



Smart Widgets: Dropdown Menu

Document Revision: 1.0
Document Date: 13th October 2017

Table of Contents

1. Description	4	5.9. Set the Min and Max Angles of Rotation.....	Error! Bookmark not defined.
2. Application Overview	4	5.10. The Need for Multiple Frames...	Error! Bookmark not defined.
3. Setup Procedure	5	5.11. Set the Minimum and Maximum Values	Error! Bookmark not defined.
4. Create a New Project.....	5	5.12. Check the Generated Frames	Error! Bookmark not defined.
5. Design the Project	5	5.13. Technique for Showing and Hiding Widget Parts	Error! Bookmark not defined.
5.1. Add a Smart Gauge Object.....	6	5.14. Add a New Layer and Swap Contents.....	Error! Bookmark not defined.
5.2. Open the Smart Widgets Editor Tool	6	5.15. Add Two Static Images to Layer 1	Error! Bookmark not defined.
5.3. Add a Face or Base Image	Error! Bookmark not defined.	5.16. Link Layer 1 and Layer 2	Error! Bookmark not defined.
5.4. Add a Manipulated Image to Layer 1.....	Error! Bookmark not defined.	5.17. Add a Background Image to Layer 3	Error! Bookmark not defined.
5.5. Rotation Point of an Image ...	Error! Bookmark not defined.	5.18. Rearrange the Layers with Respect to the Base Image	Error! Bookmark not defined.
5.6. Location of an Image in the Working Area	Error! Bookmark not defined.		
5.7. Change the Rotation Point of the Image on Layer 1	Error! Bookmark not defined.		
5.8. Center the Reference Point on the Working Area.....	Error! Bookmark not defined.		

5.19. Add a Numeric Part to Layer 2... **Error! Bookmark not defined.**

5.20. Add a Special Character to a Numerical
Part..... **Error! Bookmark not defined.**

5.21. Display Negative Values..... **Error! Bookmark not defined.**

5.22. Add and Configure an Input Object **Error! Bookmark not defined.**

6. Run the Program 14

7. Legal Notice 15

8. Contact Information 15

1. Description

This application note shows how to create a dropdown menu that overlaps a Gauge for Picaso and Diablo16 touch screen display modules.

Before getting started, the following are required:

- Any Picaso display module such as the following:

Product Name	Description
gen4-uLCD-24PT	2.4 inch resistive touch
gen4-uLCD-28PT	2.8 inch resistive touch
gen4-uLCD-32PT	3.2 inch resistive touch

- The target module can also be a Diablo16 display:

Product Name	Description
gen4-uLCD-24DT	2.4 inch resistive touch
gen4-uLCD-28DT	2.8 inch resistive touch
gen4-uLCD-32DT	3.2 inch resistive touch
gen4-uLCD-35DT	3.5 inch resistive touch
gen4-uLCD-43DT	4.3 inch resistive touch
gen4-uLCD-50DT	5.0 inch resistive touch
gen4-uLCD-70DT	7.0 inch resistive touch
gen4-uLCD-32DCT-CLB	3.2 inch capacitive touch
gen4-uLCD-35DCT-CLB	3.5 inch capacitive touch
gen4-uLCD-43DCT-CLB	4.3 inch capacitive touch
gen4-uLCD-50DCT-CLB	5.0 inch capacitive touch
gen4-uLCD-70DCT-CLB	7.0 inch capacitive touch

Visit www.4dsystems.com.au to see the latest and/or superseded 4D display module products that use the Picaso, Picaso-Lite, and Diablo16 processors. **Non-touch and SB (Super Bright) versions are also available.**

- [4D Programming Cable](#) or [uUSB-PA5](#)
- [Workshop4 IDE](#) (installed according to the installation document)

This requires the **PRO** version of Workshop4.

Note: Using a non-4D programming interface could damage the processor and void the warranty.

This application note is applicable to all touch screen 4D displays. However, Smart Gauges can also be used on non-touch displays.

2. Application Overview

The Smart Widgets Editor tool enables PRO version users to easily create custom widgets of their own design. It allows the user to create Sliders, Knobs and Gauges.

The purpose of this application note is to introduce the PRO version exclusive tool and to discuss how to create a circular progress bar using a Smart Gauge widget. This application note uses the ViSi-Genie environment.

3. Setup Procedure

For instructions on how to launch Workshop4, how to open a **ViSi-Genie** project, and how to change the target display, kindly refer to the section “**Setup Procedure**” of the application note

[First ViSi-Genie Project for Picaso](#)

or

[First ViSi-Genie Project for Diablo16](#)

4. Create a New Project

For instructions on how to create a new **ViSi-Genie** project, please refer to the section “**Create a New Project**” of the application note

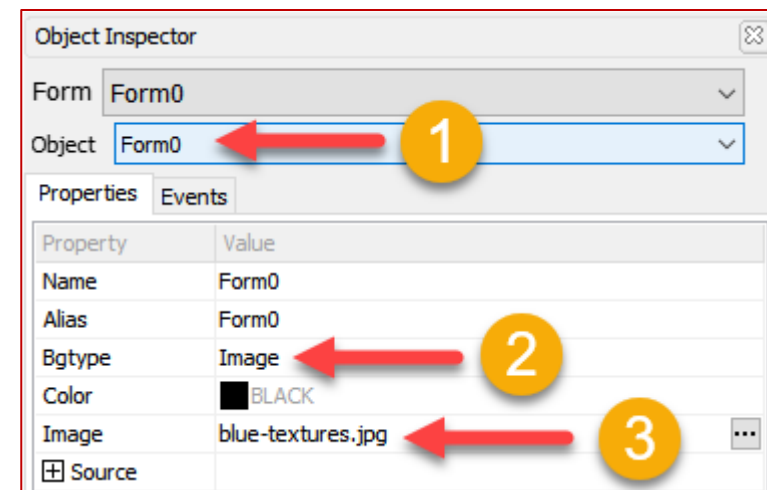
[First ViSi-Genie Project for Picaso](#)

or

[First ViSi-Genie Project for Diablo16](#)

5. Design the Project


For this application, gen4-uLCD-43DT will be used for the project. Same procedure is applicable for any Picaso and Diablo16 displays. First, set the background image of Form0.

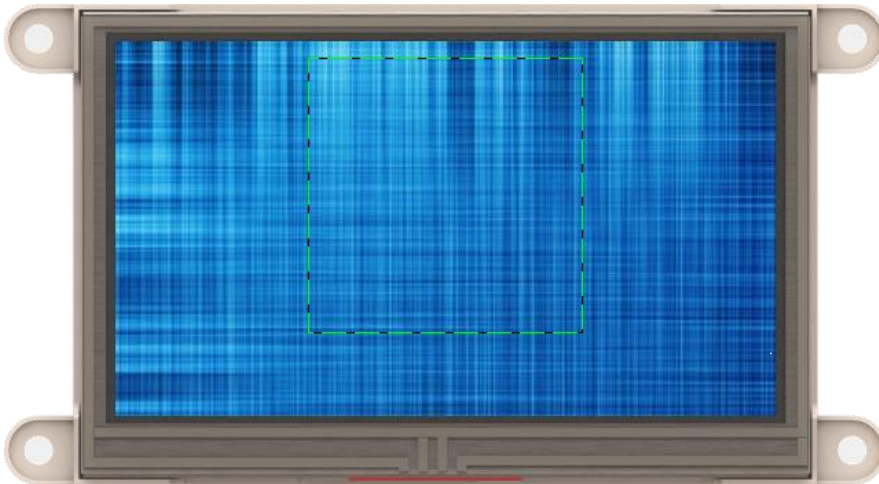


5.1. Add a Smart Gauge Object

Add a Smart Gauge widget to your ViSi-Genie project. It can be found on the Gauges tab on the Widgets Pane.





Simply click on this icon  to select it. Then place it on the WYSIWYG area.



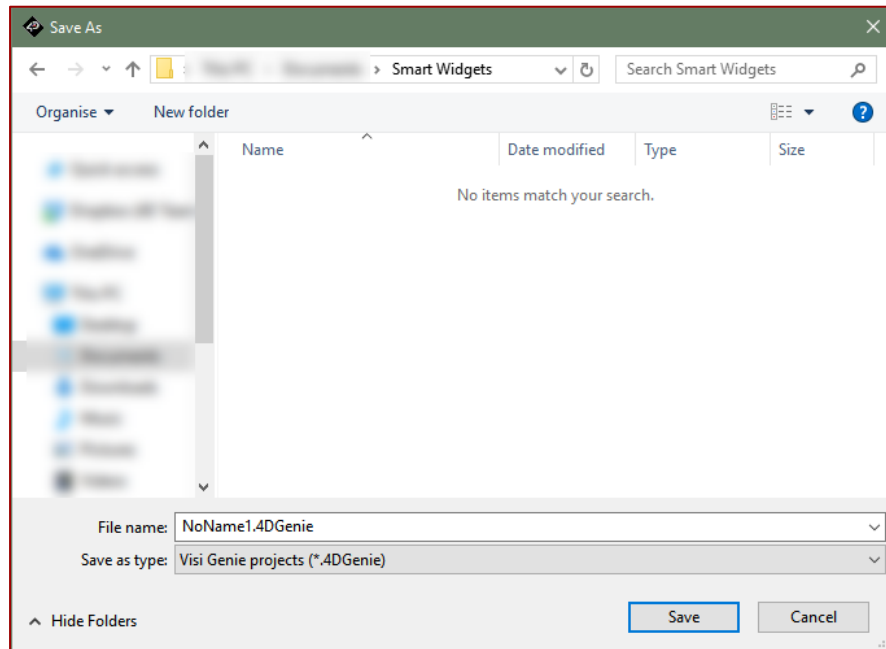
As displayed on the previous image, the widget appears empty when placed in the WYSIWYG area.

5.2. Open the Smart Widgets Editor Tool

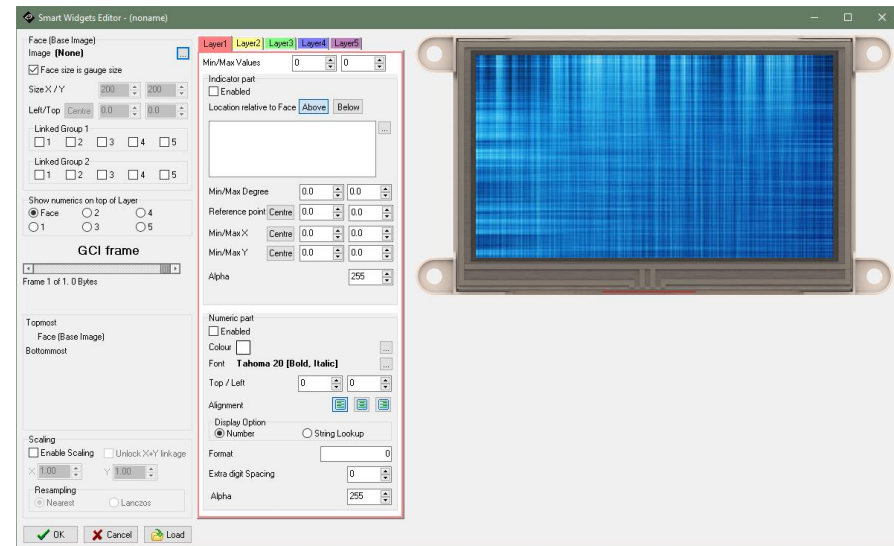
Open the Smart Widgets Editor tool by clicking on  of **Config** in the Object Inspector Properties tab.

Properties		Events
Property	Value	
Name	SmartGauge0	
Alias	SmartGauge0	
Config	SmartGauge0	
Left	140	
Top	0	

The tool requires that the project is already saved before the tool opens. Therefore, since on this case, it hasn't been saved yet, Workshop4 will automatically prompt the user to save



Save the project to desired location. The tool will open after the project has been saved.



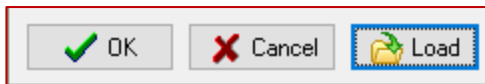
As shown in the image, this tool has a lot of parts. The next steps will focus only on the minimum tool functionalities required to make a basic circular progress bar.

For detailed discussion on how each part works, please refer to the [Smart Widgets Editor User Guide](#).

5.3. Design New Gauge or Load Saved Gauge

You may design a new Gauge or use a previously saved one before adding the dropdown menu.

Ensure that you have at least one unused layer for the Dropdown menu. For this appnote project, a previously saved gauge will be used.



The following buttons can be found on the bottom-left area of the editor window. Click on **Load**. It will automatically open to the directory of smart widgets from Workshop4.

Find **SimpleGauge.4Dsmart** and click **Open**.

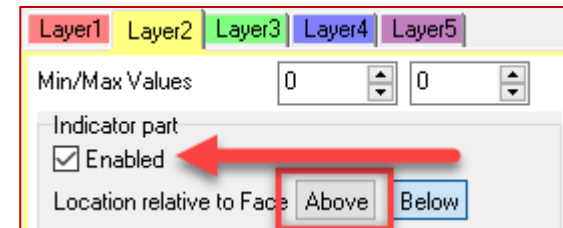


At the time, this appnote was written the gauge is still available. In case it is not available or there are any issues with the copy found, you may also find it the appnote project folder under the same filename.

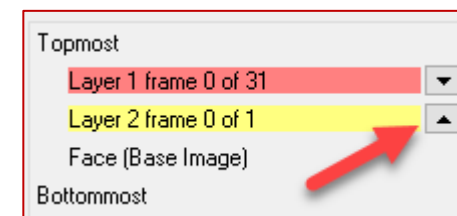
The editor should display the gauge after it loads the 4Dsmart file as shown in the previous image.

5.4. Preparing a Layer for Dropdown Menu

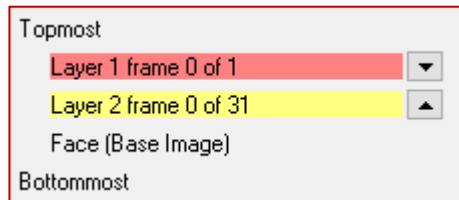
Enable another layer. This will be used for the dropdown menu. Ensure that it is the topmost layer. If you're following the same procedure as the appnote, you should be able to enable Layer 2.



Place it above the base/face image. And swap it with Layer 1.



After swapping the 2 Layers, you'll notice that the 31 frames of Layer 1 is transferred to Layer 2:

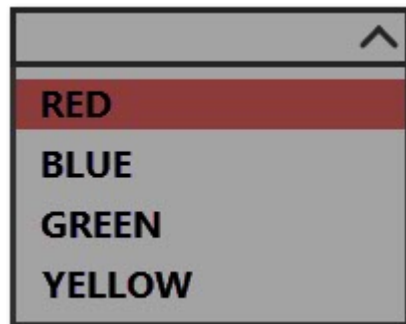


5.5. Designing the Dropdown Menu

A dropdown menu should have both close:



And open state:



Additionally, for both states, it should show the currently selected item.

For that purpose, the open dropdown menu state needs to have a frame for each item. Additionally, there needs to be multiple duplicates of close state so that it equals the number of open states.

In this appnote, the frames used for the layer are:

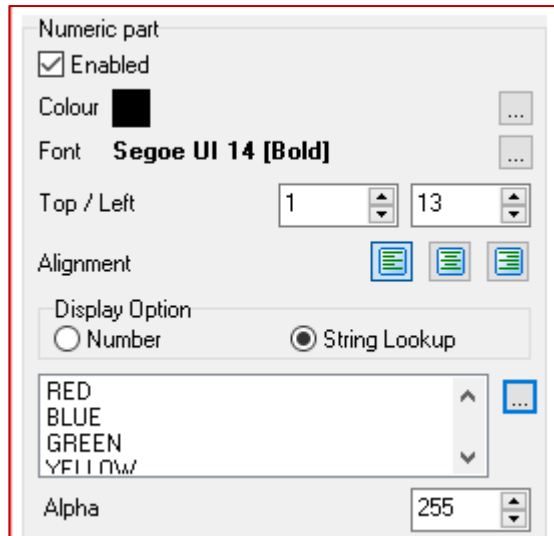
- Closed_Dropdown.png
- Closed_Dropdown.png
- Closed_Dropdown.png
- Closed_Dropdown.png
- Open_Dropdown_RED.png
- Open_Dropdown_Blue.png
- Open_Dropdown_Green.png
- Open_Dropdown_Yellow.png

Enable the Numerical Part and set it to String Lookup. The string should contain the following:

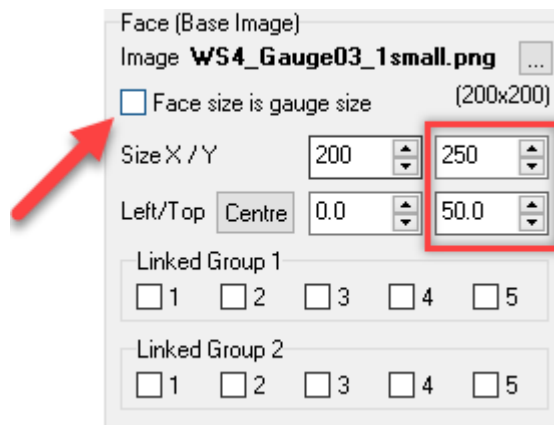
- RED
- BLUE
- GREEN
- YELLOW
- RED
- BLUE
- GREEN
- YELLOW

Position the strings correctly and choose your desired settings.

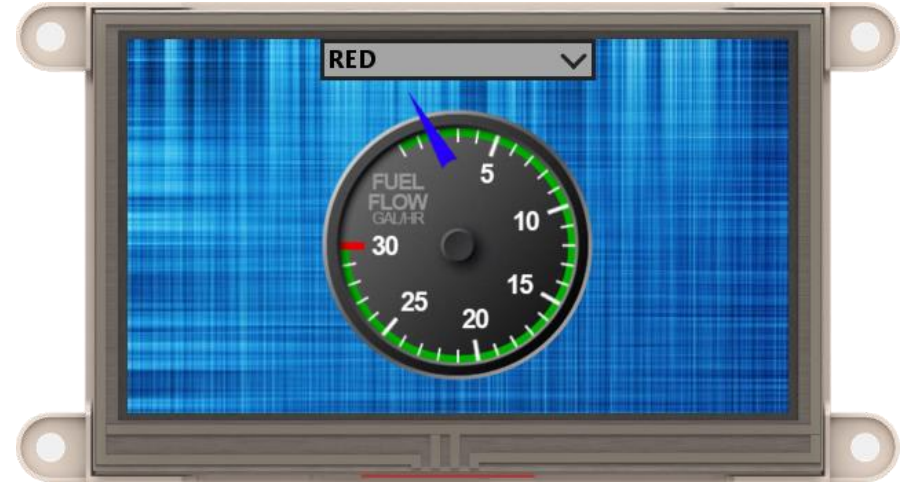
This appnote project uses the configuration below:



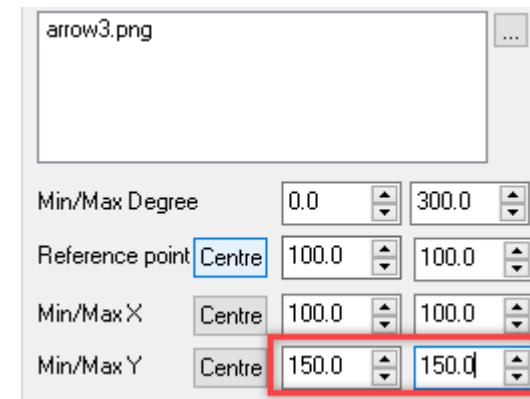
Finalize the gauge by bringing the gauge down so that when the dropdown menu is open, the gauge is not covered by it.



Adjust the face size and top position accordingly.



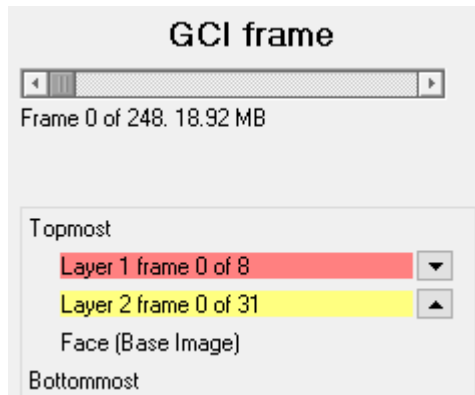
You'll notice that the needle was left behind by the base image.



Adjust the Y position accordingly.

5.6. Checking the Order of Frames

The order of frames can easily be observe by utilizing the GCI frame slider.



You can slowly observe the frame by utilizing the buttons at both ends of the slider.

The first few press on the button on the right end of the slider will show that the dropdown menu changes state from frame 0 to 7; moving a total of 8 frames. Afterwards, it will return to the layer's frame 0 (1st frame) which is when the gauge will move 1 unit.

That being said, the GCI frame can be computed as shown below:

$$frame_{GCI} = frame_{L1} + 8 (frame_{L2})$$

Where *frame* is from 0 to N.

The Layer1 frames can also be broken to 2 values.

Following the arrangement of the frames:

- Closed_Dropdown.png
- Closed_Dropdown.png
- Closed_Dropdown.png
- Closed_Dropdown.png
- Open_Dropdown_RED.png
- Open_Dropdown_Blue.png
- Open_Dropdown_Green.png
- Open_Dropdown_Yellow.png

It can be noticed that it will finish going through the 4 colors before moving to the Open state which will then go through the 4 colors then go back to the first frame.

That being the case, the formula can be expanded to:

$$frame_{GCI} = frame_{L1} + 8 (frame_{L2})$$

$$frame_{GCI} = 4 (state) + color + 8 (frame_{L2})$$

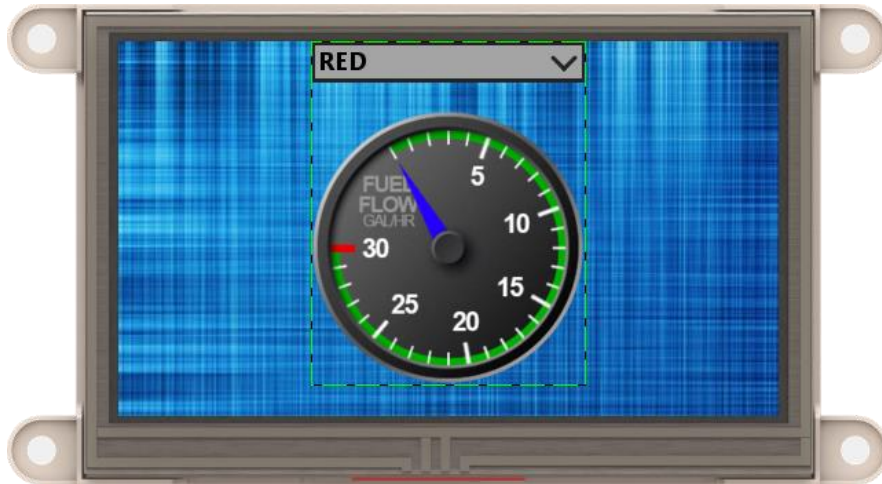
Where:

state is 0 for closed and 1 for opened dropdown menu

color is 0 to 3 for Red, Blue, Green and Yellow

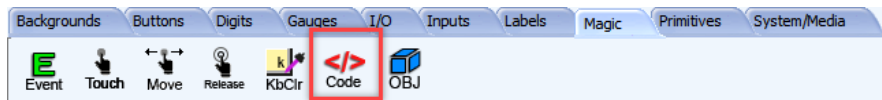
This equation will be used in the code later on.

5.7. Use SmartGauge as Input Object



SmartGauges are non-input objects by default. For this step, 4DGL code will be added to the project.

Touch functionality should be enabled for the SmartGauge object when currently at the form containing it and disabled if not. For that add a **Magic Code** to the project.



Set the insert point to be **PostActivateForm**

Property	Value
Name	MagicCode0
Alias	MagicCode0
Code	MagicCode0.inc
InsertPoint	PostActivateForm

Open the editor by clicking on  of the row **Code**

Then add the following lines of code.

```
if (CurrentForm == 0)
    img_ClearAttributes(hndl, iSmartGauge0, I_TOUCH_DISABLE);
else
    img_SetAttributes(hndl, iSmartGauge0, I_TOUCH_DISABLE);
endif
```

Add another **Magic Code** and set its insertion point to **Global**

Property	Value
Name	MagicCode1
Alias	MagicCode1
Code	MagicCode1.inc
InsertPoint	Constant/Global/Data

```
var gaugeValue[4];
var selectedItem;
var menuState;
```

Insert the above lines of code to declare the following variables and array. Notice that they are values relating to the frames formula.

We need to program what it will do when a touch event occurs. Add **Magic Touch**, **Magic Release** and **Magic Move** objects to the project. Then add the following lines of code to each of the three.

```
if (ImageTouched == iSmartGauge0)
    ImageTouched := -1;
endif
```

Open **Magic Touch** object add these lines of code inside the **if** condition.

```
var item, topPos;
topPos := img_GetWord(hndl, iSmartGauge0, IMAGE_YPOS);
item := (TouchYpos - topPos) / 32 - 1;

if (item < 0)
    menuState := !menuState;
else if (menuState && item < 4)
    selectedItem := item;
endif

updateGauge();
```

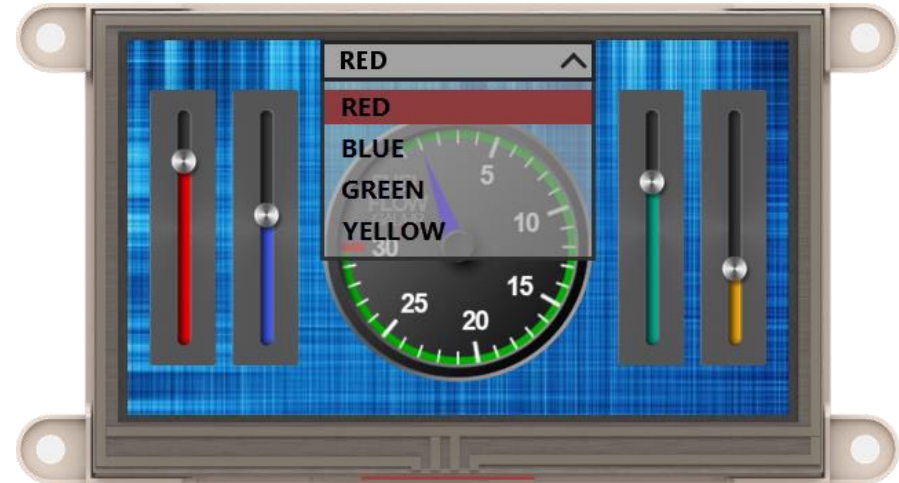
As you may have noticed, there is a function used in the above code which we haven't defined yet. Go back to the **Global Magic Code** and add these lines.

```
func updateGauge()
    var frame, gaugeVal;
    gaugeVal := gaugeValue[selectedItem];
    frame := 4 * menuState + selectedItem + 8 * gaugeVal;

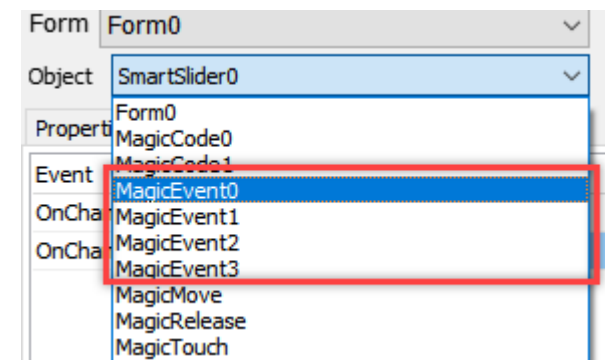
    WriteObject(tSmartGauge, 0, frame);
Endfunc
```

5.8. Add Multiple Input Objects

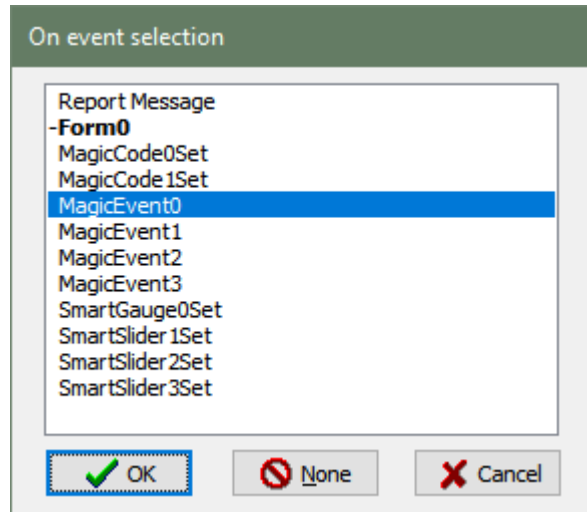
Add four slider or knob widgets to the project. The project included with this appnote uses four SmartSliders.



Add a Magic Event for each of the four input objects.



Set the **OnChanging** event of each input object to a Magic Event.



Ensure that each of them has unique **Magic Events**

For each of the **Magic Event** objects. Insert these lines of code inside the function.

```
gaugeValue[n] := newval;  
updateGauge();
```

You need to replace ***n*** with 0 to 3 so it is unique for each **Magic Events**

6. Run the Program

For instructions on how to save a **ViSi-Genie** project, how to connect the target display to the PC, how to select the program destination, and how to compile and download a program, please refer to the section “**Run the Program**” of the application note

[First ViSi-Genie Project for Picaso](#)

Or

[First ViSi-Genie Project for Diablo16](#)

7. Legal Notice

Proprietary Information

The information contained in this document is the property of 4D Labs Semiconductors and may be the subject of patents pending or granted, and must not be copied or disclosed without prior written permission.

4D Labs Semiconductors endeavours to ensure that the information in this document is correct and fairly stated but does not accept liability for any error or omission. The development of 4D Labs Semiconductors products and services is continuous and published information may not be up to date. It is important to check the current position with 4D Labs Semiconductors. 4D Labs Semiconductors reserves the right to modify, update or make changes to Specifications or written material without prior notice at any time.

All trademarks belong to their respective owners and are recognised and acknowledged.

Disclaimer of Warranties & Limitation of Liability

4D Labs Semiconductors makes no warranty, either expressed or implied with respect to any product, and specifically disclaims all other warranties, including, without limitation, warranties for merchantability, non-infringement and fitness for any particular purpose.

Information contained in this publication regarding device applications and the like is provided only for your convenience and may be superseded by updates. It is your responsibility to ensure that your application meets with your specifications.

Images and graphics used throughout this document are for illustrative purposes only. All images and graphics used are possible to be displayed on the 4D Labs Semiconductors range of products, however the quality may vary.

In no event shall 4D Labs Semiconductors be liable to the buyer or to any third party for any indirect, incidental, special, consequential, punitive or exemplary damages (including without limitation lost profits, lost savings, or loss of business opportunity) arising out of or relating to any product or service provided or to be provided by 4D Labs Semiconductors, or the use or inability to use the same, even if 4D Labs Semiconductors has been advised of the possibility of such damages.

4D Labs Semiconductors products are not fault tolerant nor designed, manufactured or intended for use or resale as on line control equipment in hazardous environments requiring fail – safe performance, such as in the operation of nuclear facilities, aircraft navigation or communication systems, air traffic control, direct life support machines or weapons systems in which the failure of the product could lead directly to death, personal injury or severe physical or environmental damage ('High Risk Activities'). 4D Labs Semiconductors and its suppliers specifically disclaim any expressed or implied warranty of fitness for High Risk Activities.

Use of 4D Labs Semiconductors' products and devices in 'High Risk Activities' and in any other application is entirely at the buyer's risk, and the buyer agrees to defend, indemnify and hold harmless 4D Labs Semiconductors from any and all damages, claims, suits, or expenses resulting from such use. No licenses are conveyed, implicitly or otherwise, under any 4D Labs Semiconductors intellectual property rights.

8. Contact Information

For Technical Support: www.4dlabs.com.au/support

For Sales Support: sales@4dlabs.com.au

Website: www.4dlabs.com.au

Copyright 4D Labs Semiconductors 2000-2017.