



ViSi Genie Tank

DOCUMENT DATE: **9th May 2019**
DOCUMENT REVISION: **1.1**



Description

This application note provides a first hands-on example with ViSi-Genie and describes all the steps related to a project.

Before getting started, the following are required:

- Workshop 4 has been installed according to the document Workshop 4 Installation.
- The user is familiar with the Workshop 4 environment and with the fundamentals of ViSi-Genie, as described in Workshop 4 User Guide and ViSi-Genie User Guide.
- When downloading an application note, a list of recommended application notes is shown. It is assumed that the user has read or has a working knowledge of the topics discussed in these recommended application notes.

Content

Description	2
Content	2
Application Overview	3
Setup Procedure	4
Create a New Project	4
<i>Create a New Project</i>	<i>4</i>
Design the Project.....	5
<i>Adding a Tank.....</i>	<i>5</i>
<i>Adding LED Digits.....</i>	<i>8</i>
<i>Adding a Slider.....</i>	<i>9</i>
<i>Configuring the Slider.....</i>	<i>10</i>
The OnChanged Event	10
Linking Objects	11
<i>Adding a Scale.....</i>	<i>11</i>
<i>Adding a Static Text.....</i>	<i>13</i>
Build and Upload the Project.....	14
Write to Tank Object.....	14
<i>Use the GTX Tool to Analyse the Messages.....</i>	<i>14</i>
Launch the GTX Tool	15
<i>The Tank.....</i>	<i>15</i>
Change the Status of the Tank	15
Interrogate the Display for the Status of the Tank	16

Proprietary Information 18

Disclaimer of Warranties & Limitation of Liability..... 18

Application Overview

It is often difficult to design a graphical display without being able to see the immediate results of the application code. ViSi-Genie is the perfect software tool that allows users to see the instant results of their desired graphical layout with this large selection of gauges and meters (called objects or widgets). The user can simply click on the desired widget to select it and click on the simulated display to place the widget. The following are examples of widgets used in this application note.



Slider



Tank



Scale



LED Digits

The project developed in this application note demonstrates basic touch functionality and object interaction. The user moves or touches the slider, and the tank and LED digits change their values to correspond with the slider's change in status. By default, input objects such as the slider respond to touch. The user can configure an input object to drive an output object such as the tank or the LED digits.

Setup Procedure

For instructions on how to launch Workshop 4, 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:

[ViSi Genie Getting Started – First Project for Picaso Displays](#) (for Picaso)

or

[ViSi Genie Getting Started – First Project for Diablo16 Displays](#) (for Diablo16).

Create a New Project

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

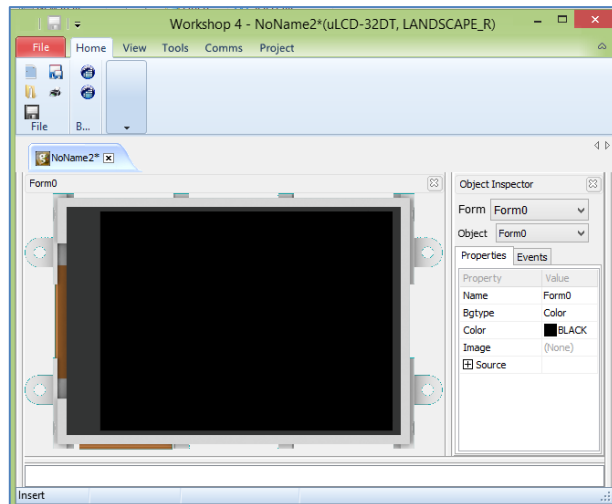
[ViSi Genie Getting Started – First Project for Picaso Displays](#) (for Picaso)

or

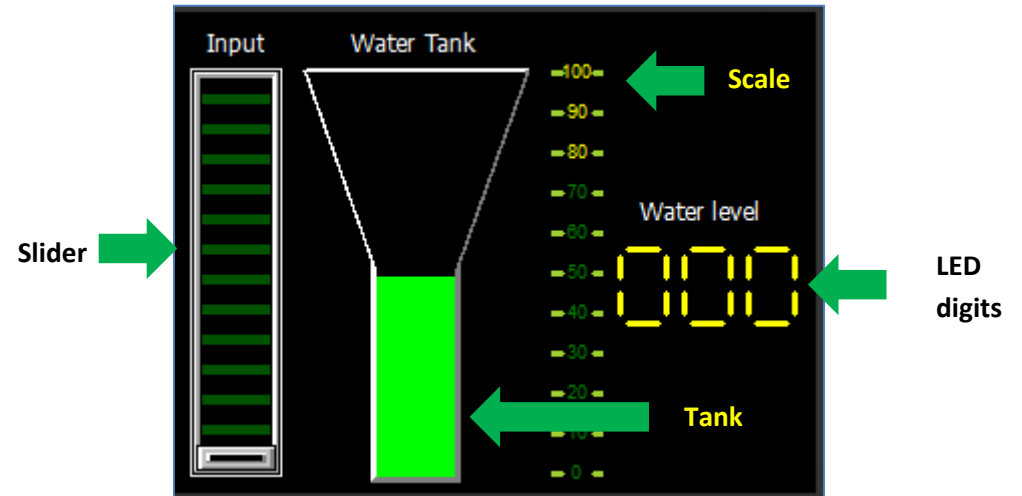
[ViSi Genie Getting Started – First Project for Diablo16 Displays](#) (for Diablo16).

Design the Project

Everything is now ready to start designing the project. **Workshop 4** displays an empty screen, called **Form0**. A **form** is like a page on the screen. The form can contain **widgets** or **objects**, like trackbars, sliders, displays or keyboards. Below is an empty form.



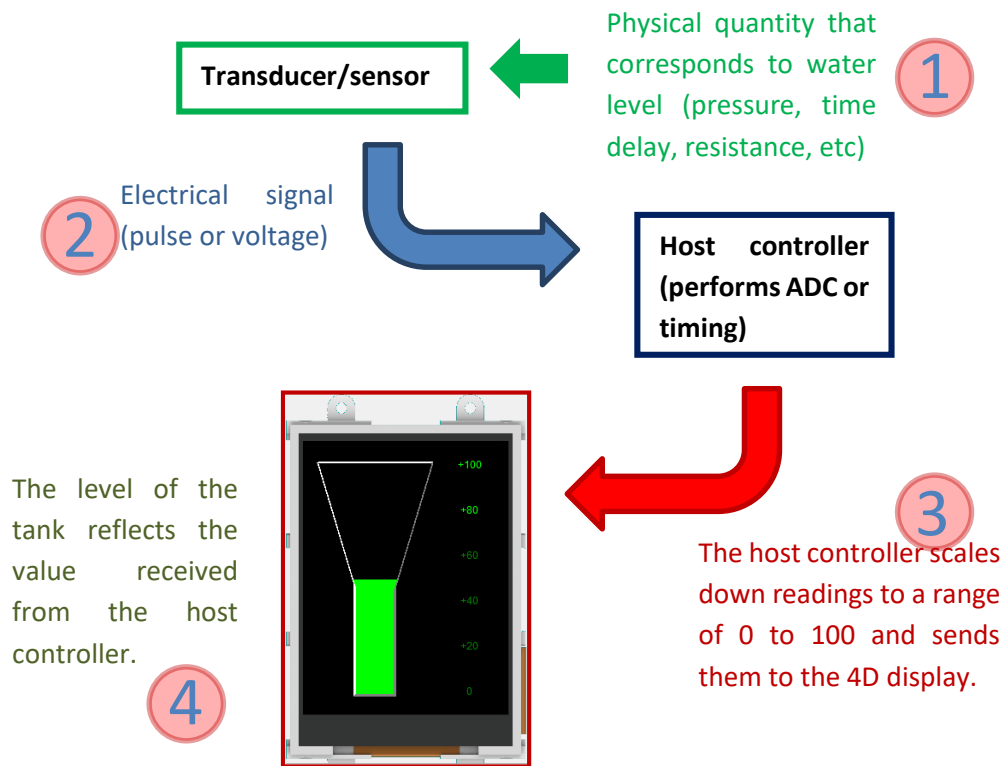
At the end of this section, the user will be able to create a form with four objects. The final form will look like as shown below, with the labels excluded.



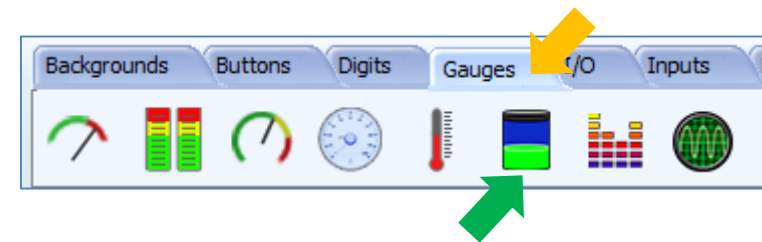
The procedure for adding each of these objects will now be discussed.

Adding a Tank

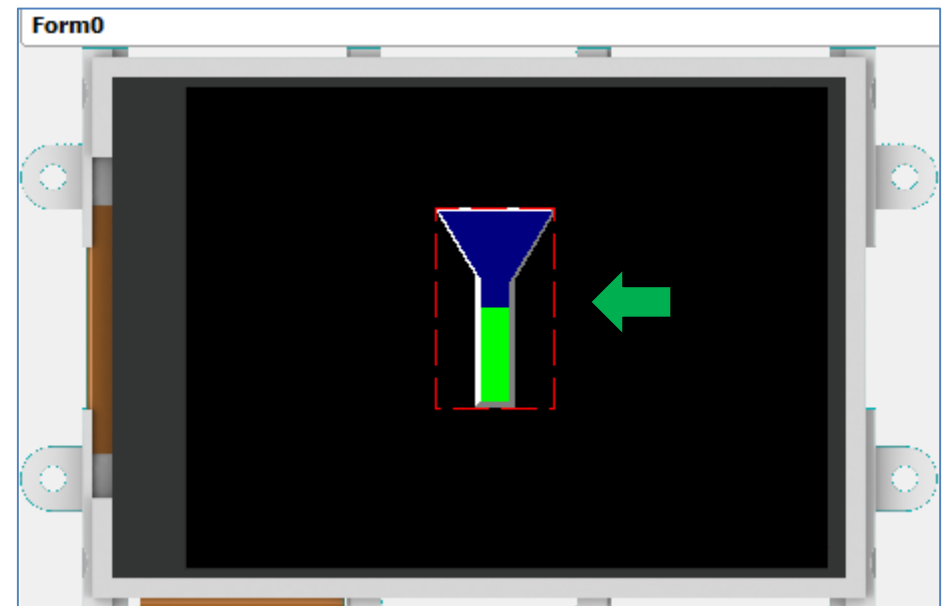
The tank can be used to visually represent how much of a container is occupied. In this example, the tank is used as a water level indicator. In real world applications, water level can be determined using various kinds of sensors - ultrasonic, resistive, pressure, etc. Through ADC or use of timers, a host controller can convert the output signal of these sensors (pulse or usually voltage) to binary levels or time intervals. These values can be further scaled or translated to values within a range of 0 to 100, which can be sent to a 4D display module to drive the tank object. To illustrate:



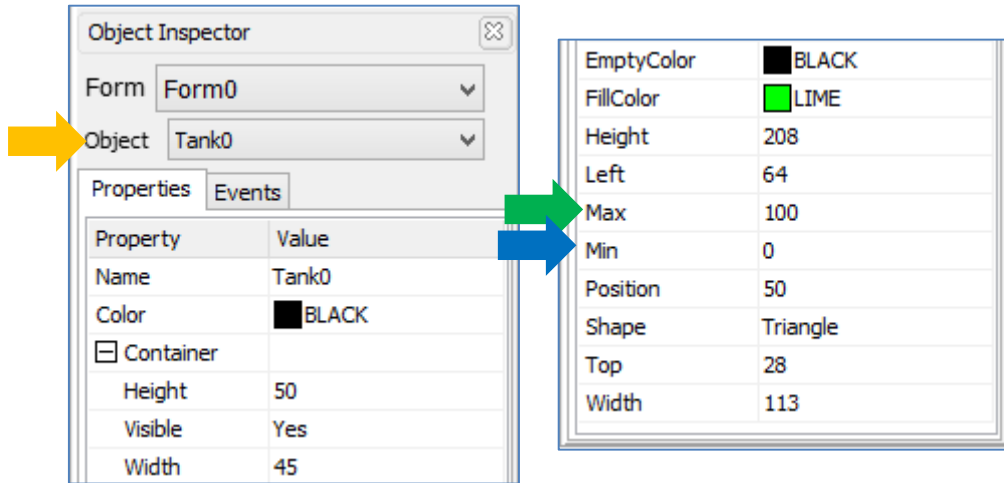
In this project, the values to be received from an external host controller will be simulated using a slider instead. The slider, when moved, will send values to the tank. The tank will then change its level or value. To add a tank, go to the **Gauges** pane then click on the **tank** icon.



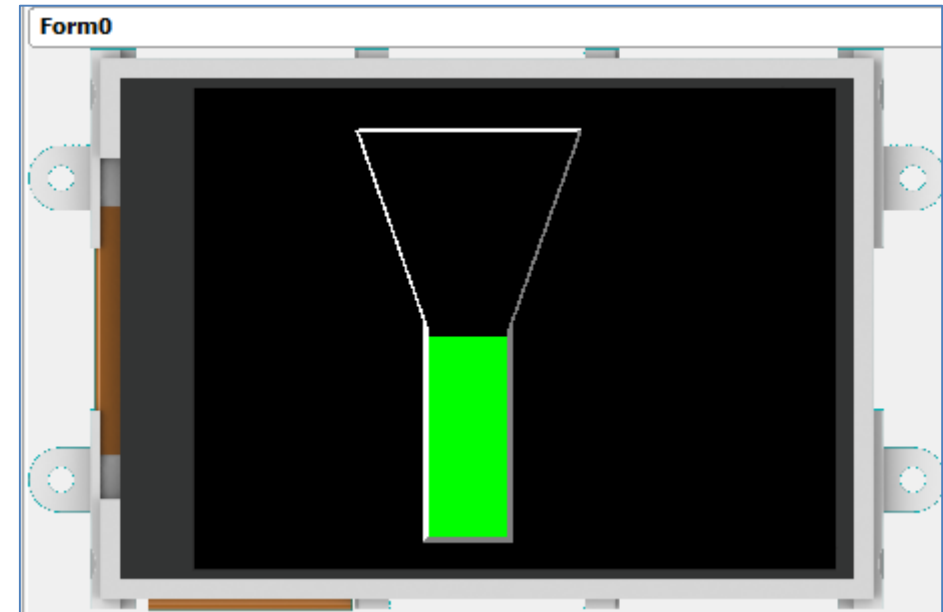
Click on the **WYSIWYG** (What-You-See-Is-What-You-Get) screen to put the tank in place. The WYSIWYG screen simulates the actual appearance of the display module screen.



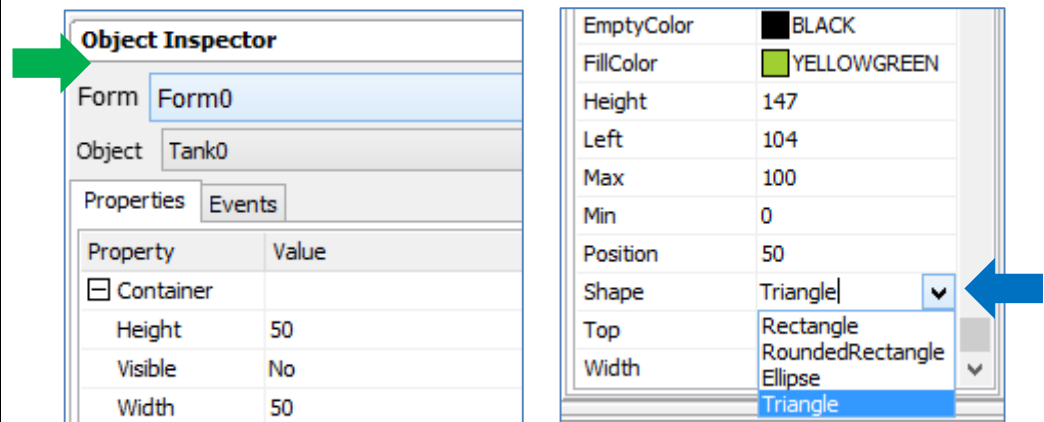
The object can be dragged to any desired location and resized to the desired dimensions. The **Object Inspector** on the right part of the screen displays all the properties of the newly created tank object named **Tank0**.

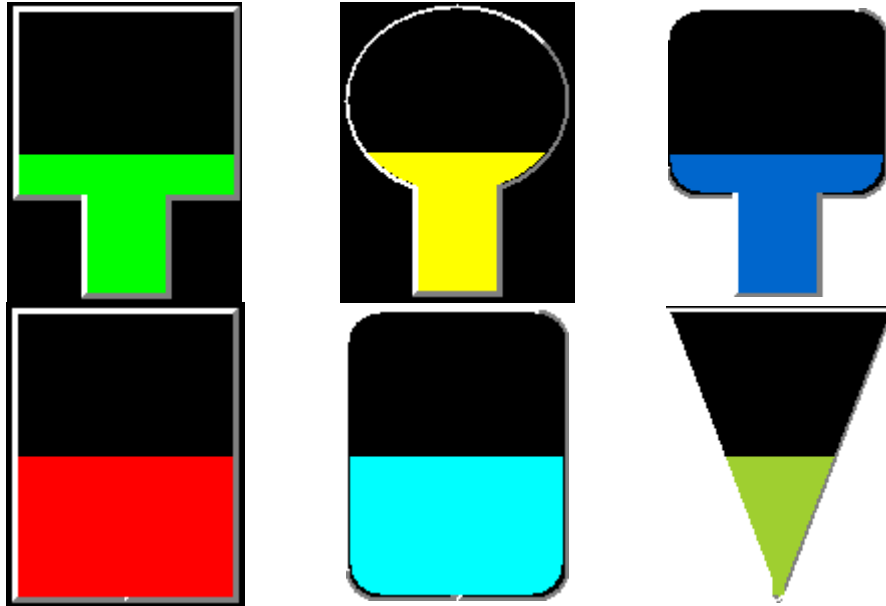


Take time to experiment with the different properties. Take note of the maximum and minimum values. These will correspond to the maximum and minimum values of the slider. The final appearance of the tank used in this project is shown below.



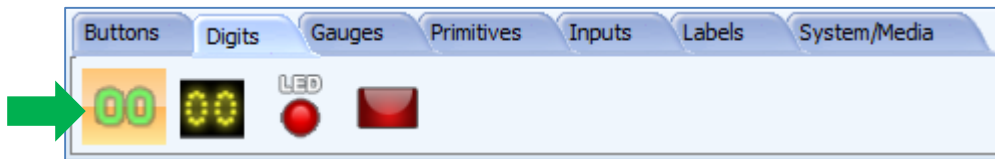
The shape and containers properties can also be edited to come up with different tank shapes.



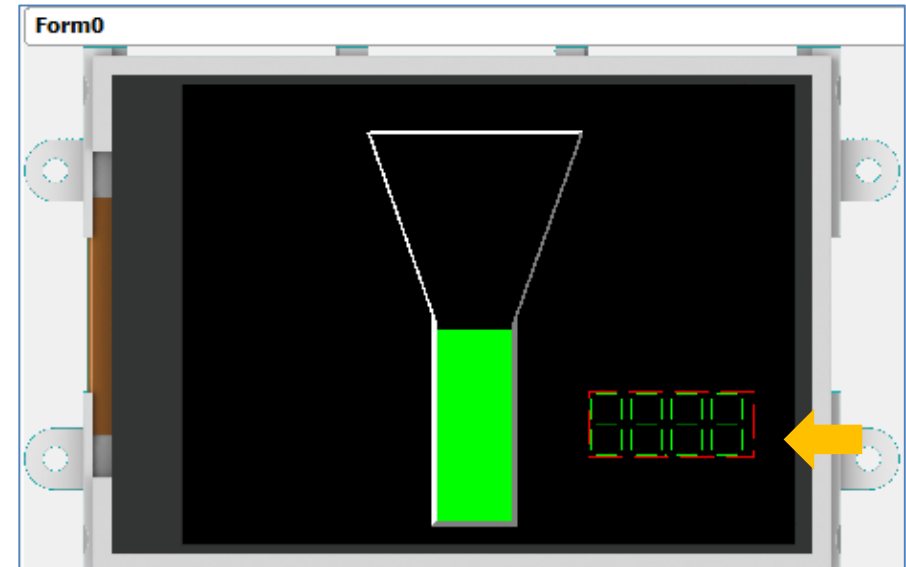


Adding LED Digits

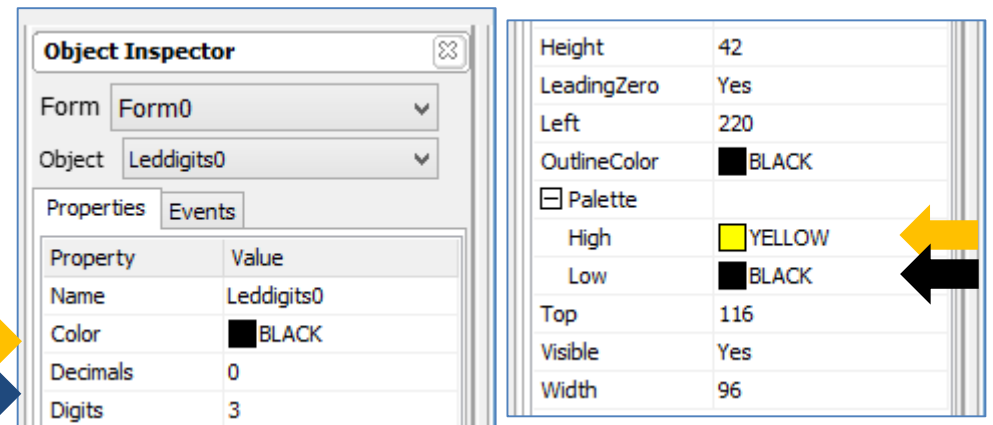
The **LED digits** object updates its value when the value of Tank0 has changed. To add a LED digits object, go to the **Digits** pane and select the first icon.



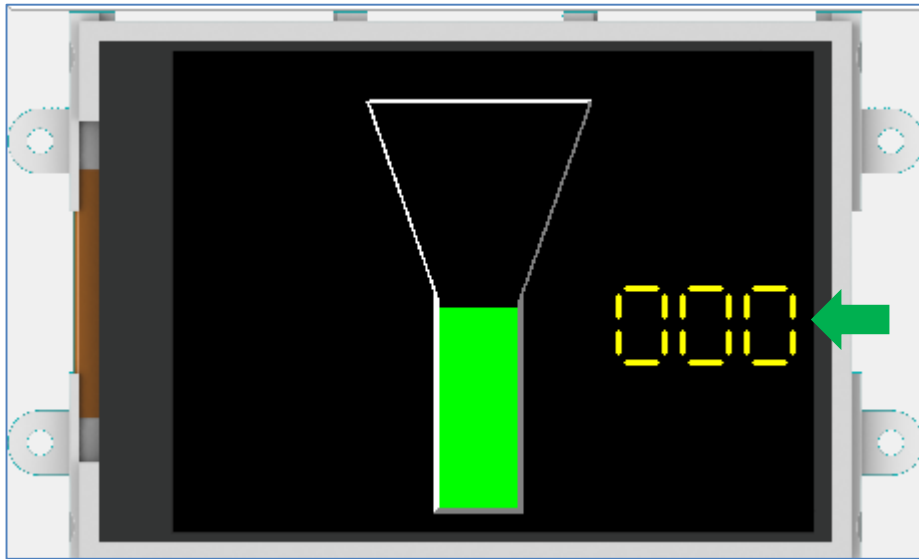
Click on the WYSIWYG screen to place the object.



Go to the Object inspector and set the following property values.

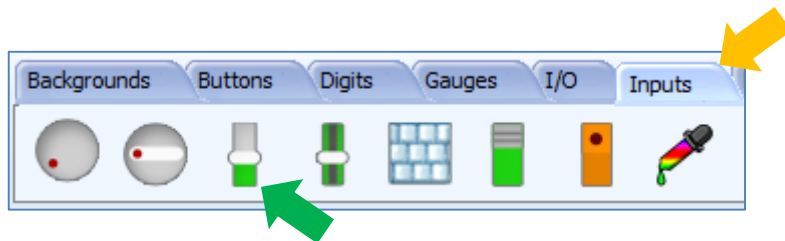


The updated appearance of the LED digits object is shown below.

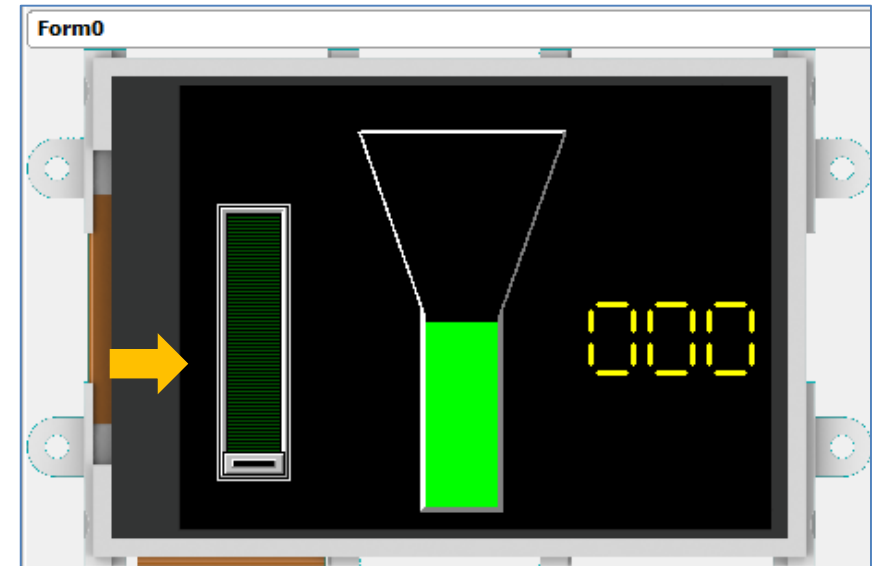


Adding a Slider

The slider responds to the user's touch and drives the tank and LED digits. To add a slider, go to the **Inputs** pane and click on the **slider** icon.



Click on the WYSIWYG screen to place the slider. Drag the object to any desired location.

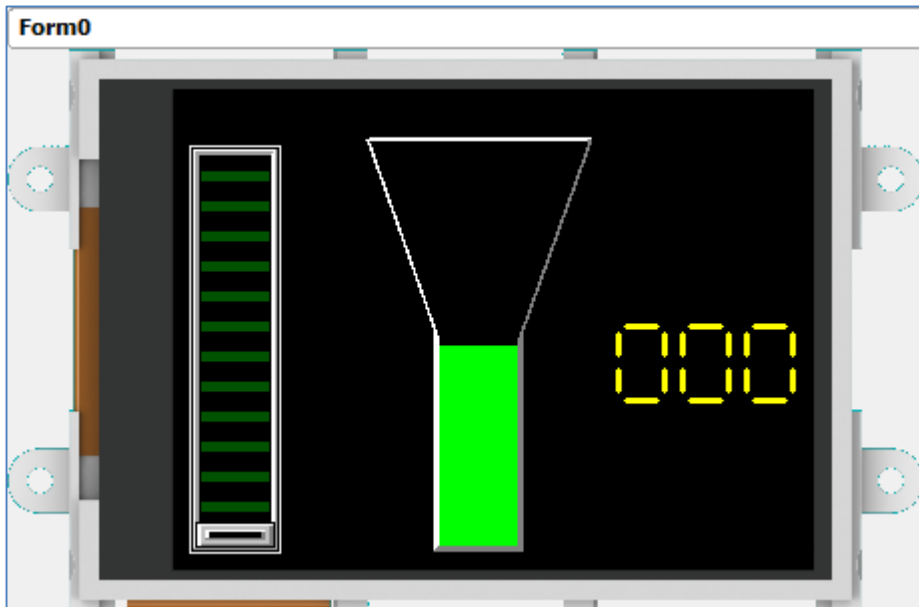


In the **Object Inspector**, the minimum value is 0 and maximum value is 100 by default. The slider in this project has the following properties.

Object Inspector	
Form	Form0
Object	Slider0
Properties	
Property	Value
Name	Slider0
Bevel	<input checked="" type="checkbox"/>
BorderColor	cBtnFace
BorderWidth	1
Color	BLACK
Height	204


Left	8
Maxvalue	100
Minvalue	0
Orientation	Vertical
Palette	
High	YELLOW
Low	0x005100
Spacing	10
SolidFill	Yes
TickWidth	5
Top	28
Visible	Yes
Width	46

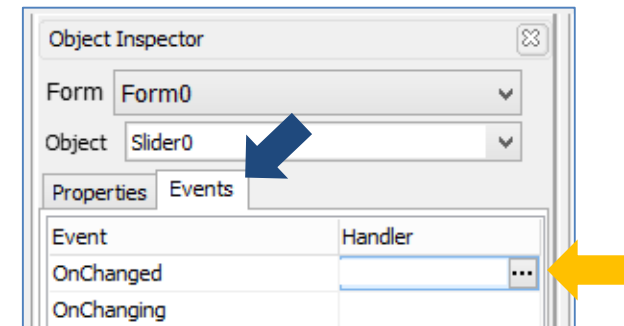
The final appearance of the slider is shown below.



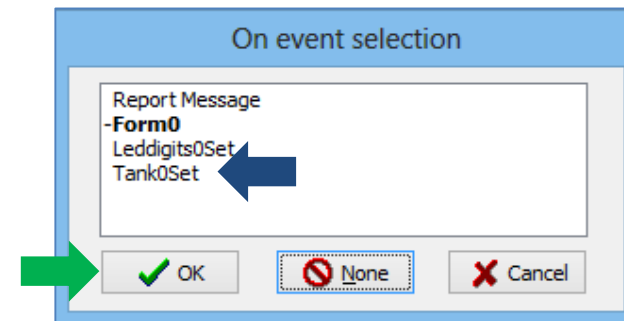
Configuring the Slider

The OnChanged Event

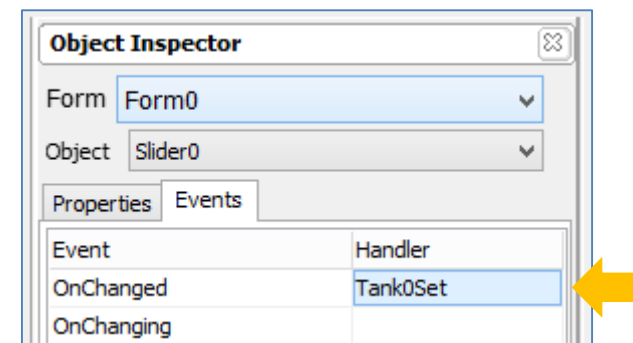
An input object such as the slider can be configured to change the status of another object when its status has changed. To do this, click on the Events tab in the object inspector and click on the  symbol in the **OnChanged** line.



The **On event selection** window appears. Select **Tank0Set** and click **OK**.




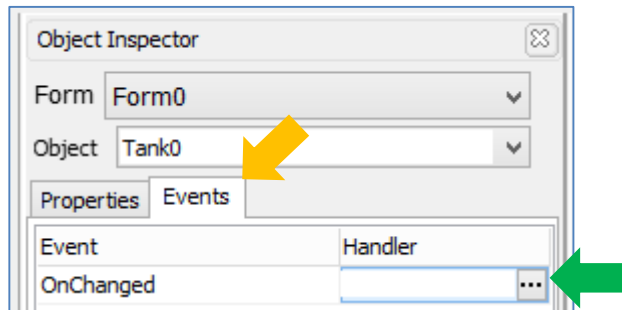
The Events pane is now updated.



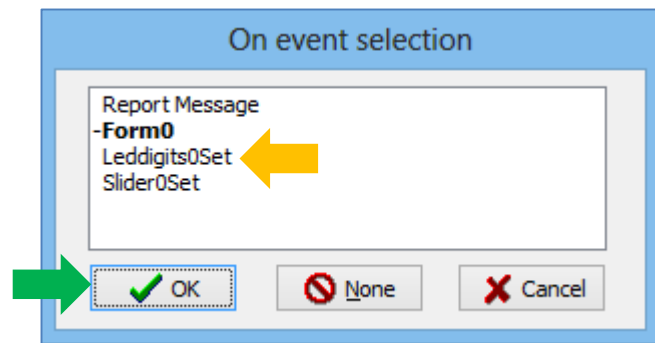
Now when Slider0 has moved, it sends its value to Tank0. Tank0 then change its status to reflect the value received from Slider0.

Linking Objects

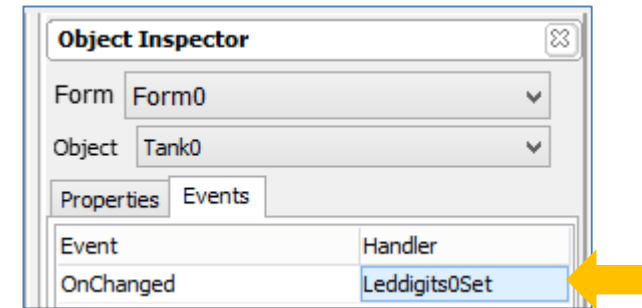
After having linked Slider0 to Tank0, it is also possible to further extend the link by configuring Tank0 to send its value to Leddigits0. To do this, click on Tank0 to select it. Select the Events tab in the object inspector and click on the  symbol in the **OnChanged** line.



The **On event selection** window appears. Select **Leddigit0Set** and click **OK**.



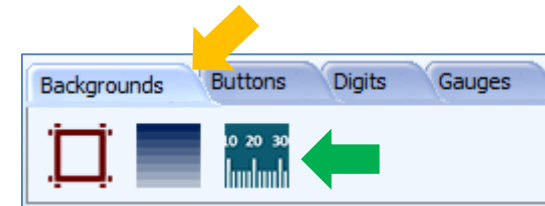
The Events pane is now updated.



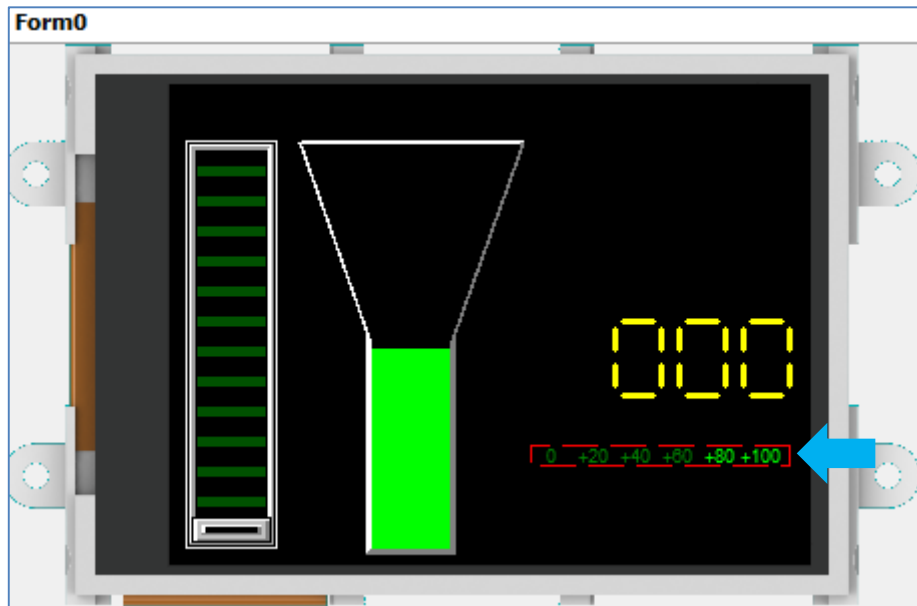
Now when Slider0 has moved, it sends its value to Tank0. Tank0 then changes its status and sends a value to Leddigits0. Leddigits0 receives and displays the value from Tank0. To learn more about how objects are linked and classified (input/output/combined), refer to the [ViSi Genie User Guide](#).

Adding a Scale

To add a scale, go to the **Backgrounds** pane and click on the **scale** icon.



Click on the WYSIWYG screen to place the object.

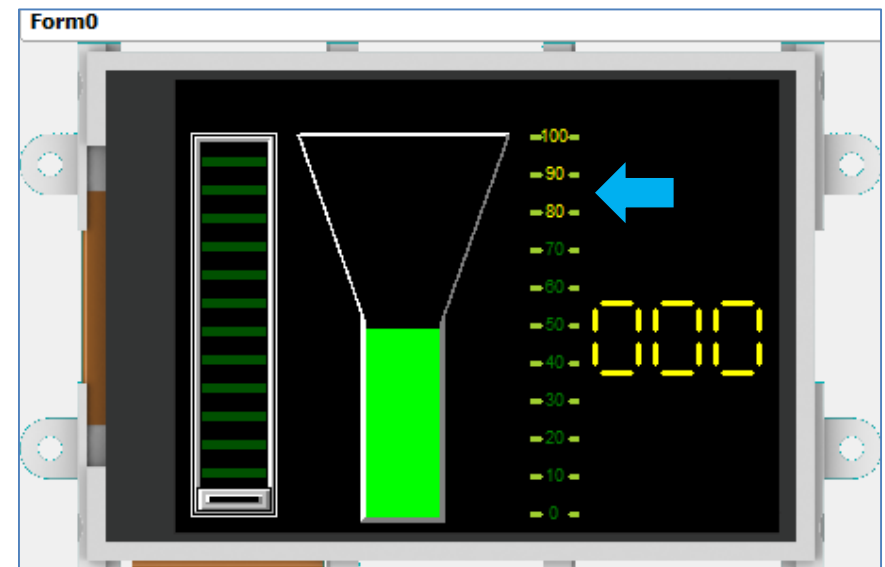


The **Object Inspector** on the right part of the screen displays all the properties of the newly created scale named **Scale0**. The scale object used in this project has the following properties.

Object Inspector	
Form	Form0
Object	Scale0
Properties Events	
Property	Value
Name	Scale0
Alignment	Center
Color	BLACK
Digits	3
Font	(GREEN, [], Arial, 7, [])
Height	221
Layout	Center
LeadingZero	No
Left	188

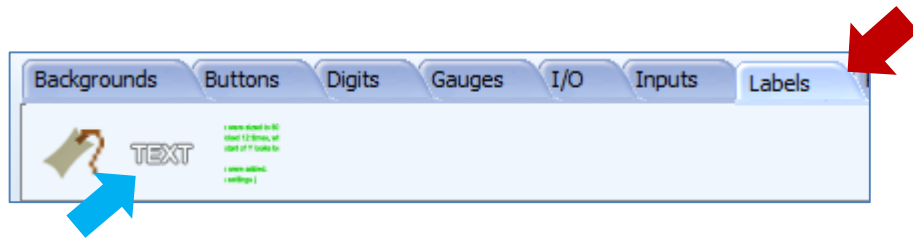
Maxvalue	100
Minvalue	0
Orientation	Vertical
PeakColor	YELLOW
PeakLevel	80
Scalecolor	YELLOWGREEN
ScaleOffset	0
ShowSign	No
TickMarks	Both
Ticks	10
TicksHeight	5
TicksWidth	3
Top	19
Transparent	Yes
Width	26

Shown below is the final appearance of the scale.

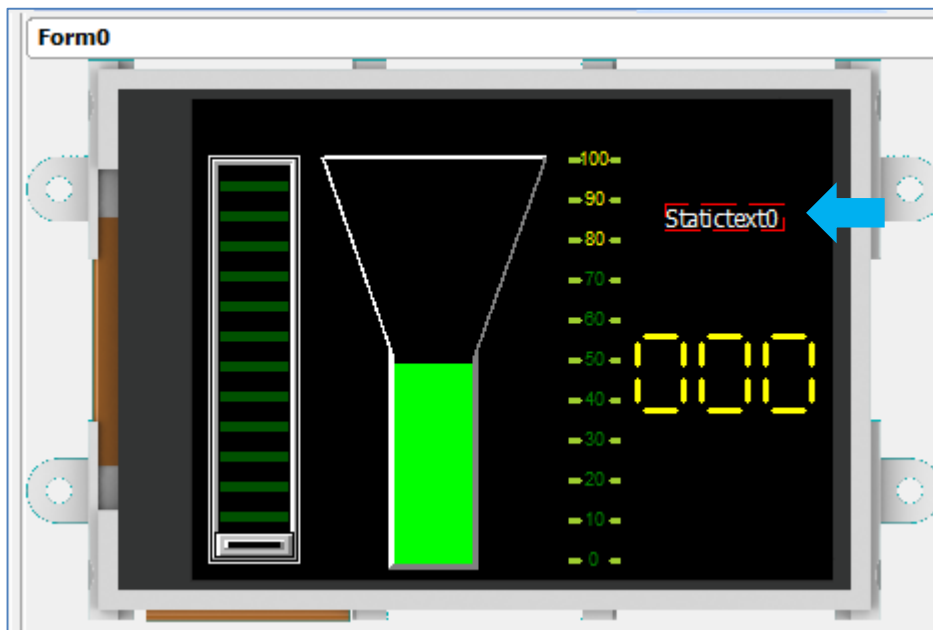


Adding a Static Text

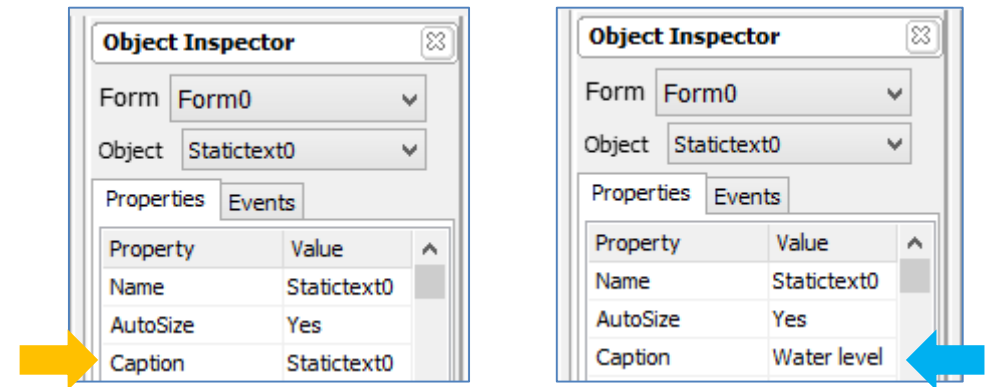
To add a static text, go to the **Labels** pane and click on the **static text** icon.



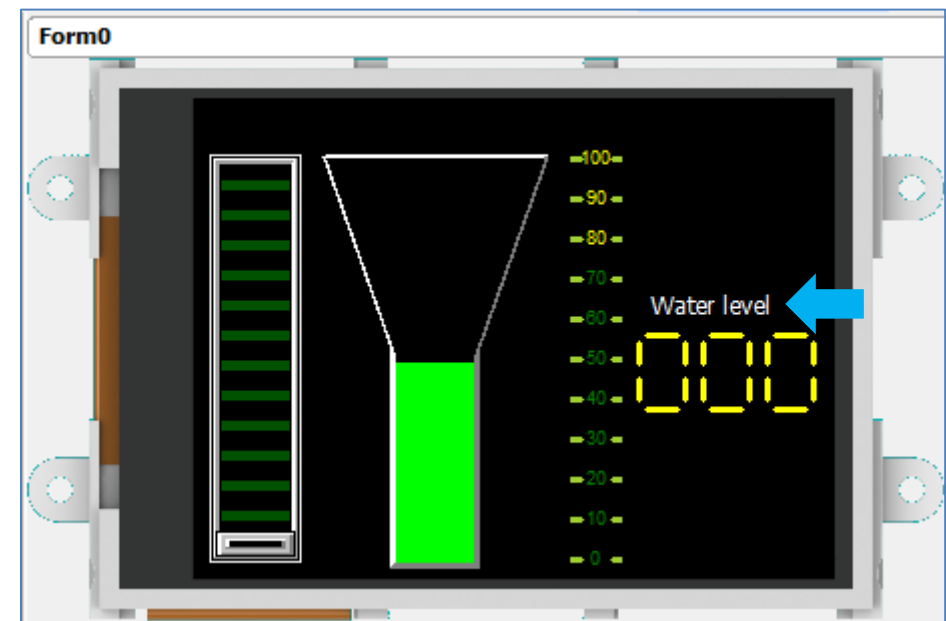
Click on the WYSIWYG screen to place the object. Drag it to any desired location.



Change the caption in the object inspector.



The WYSIWYG screen is updated accordingly.



Now add two more static text objects – “Input” and “Water Tank”.

Build and Upload the Project

For instructions on how to build and upload a ViSi-Genie project to the target display, please refer to the section “**Build and Upload the Project**” of the application note

[ViSi Genie Getting Started – First Project for Picaso Displays](#) (for Picaso)

or

[ViSi Genie Getting Started – First Project for Diablo16 Displays](#) (for Diablo16).

The uLCD-32PTU and/or the uLCD-35DT display modules are commonly used as examples, but the procedure is the same for other displays.

Write to Tank Object

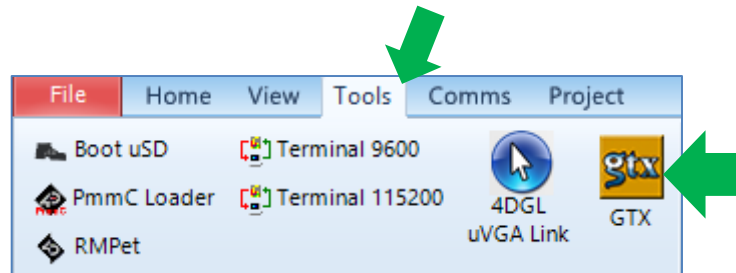
The slider sends values to the tank to simulate the values to be received from an external host controller. Being without a host, this application is a stand-alone project with input and output objects. However, it can be made to work with an external host controller thru serial communication. The host controller will send messages to the display module to control the status of the tank. The communication between the display module and the host controller has a protocol (called the ViSi-Genie Communications Protocol) which defines the format of the messages sent and received. This section explains to the user how to interpret these messages. An understanding of this section is necessary for users who intend to interface the display to a host. The [ViSi Genie Reference Manual](#) is recommended for advanced users.

Use the GTX Tool to Analyse the Messages

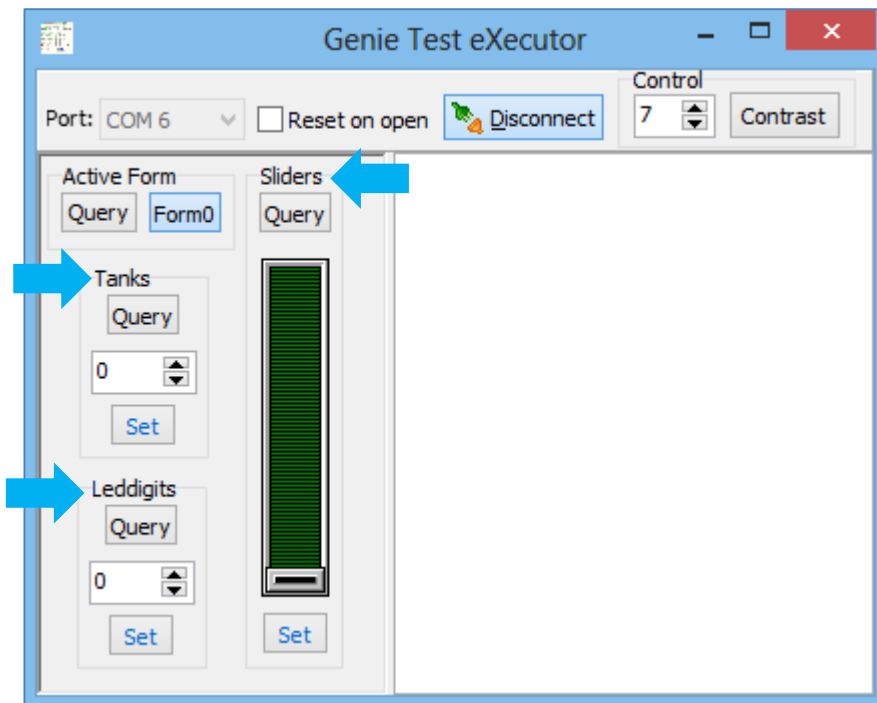
Using the GTX or **Genie Test eXecutor** tool is the first option to get the messages sent by the screen to the host. Here the PC will be the host. The GTX tool is a part of the Workshop 4 IDE. It allows the user to receive, observe, and send messages from and to the display module. It is an essential debugging tool.

Launch the GTX Tool

Under Tools menu click on the GTX tool button.



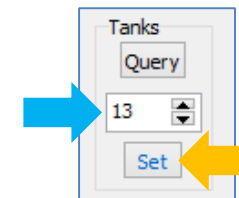
A new window appears, with the form and objects created previously.



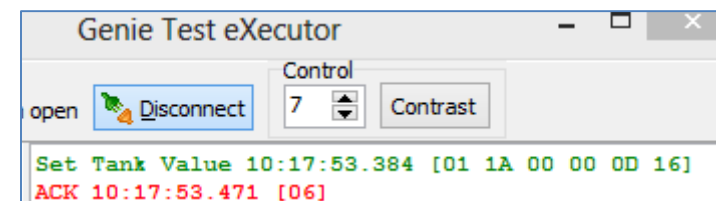
The Tank

Change the Status of the Tank

In the GTX tool window, use the up/down buttons of the tank object or type in the box an integer between 0 and 100 and press **Set**. On the display module, note that the level of the tank has changed.



Also, messages are sent to and received from the display module.



The white area on the right displays

- in **green** the messages sent to the display module
- and in **red** the messages received from the display module

The actual message bytes are those inside the brackets. These values are in hexadecimal. The figure preceding the actual message is the computer time at which the message is sent. A label is also included to tell the observer what the message represents.

```

time
message
Set Tank Value 10:17:53.384 [01 1A 00 00 0D 16]
    
```

The message sent is formatted according to the following pattern:

Command	Object Type	Object Index	Value MSB	Value LSB	Checksum
01	1A	00	00	0D	16
WRITE_OBJ	Tank	First	0x000D		

The message stands for “Write to the first tank object on the display module the value 0x000D.” Converting the hexadecimal value 0x000D to decimal will yield the value 13.

The checksum is a means for the host to verify if the message received is correct. This byte is calculated by XOR’ing all bytes in the message from (and including) the CMD or command byte to the last parameter byte. Then, the result is appended to the end to yield the checksum byte. If the message is correct, XOR’ing all the bytes (including the checksum byte) will give a result of zero. Checking the integrity of a message using the checksum byte shall be handled by the host.

ACK = 0x06 as shown below

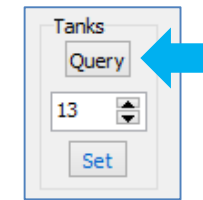
```

ACK 11:55:48.098 [06]
    
```

is an acknowledgment from the display module which means that it has understood the message.

Interrogate the Display for the Status of the Tank

The PC can ask the display module for the value of the tank by sending a message. Now on the display module randomly move the slider to drive the tank. In the GTX tool window press Query.



Observe the message area.

```

Request Tank Value 10:46:56.073 [00 1A 00 1A]
Tank Value 10:46:56.124 [05 1A 00 00 4F 50]
    
```

The PC sends a request message. The display module replies with the current value of the tank. The messages sent and received are formatted according to the following patterns.

Command	Object Type	Object Index	Value MSB	Value LSB	Checksum
00	1A	00	-	-	1A
READ_OBJ	Tank	First	Not applicable		
05	1A	00	00	4F	50
REPORT_OBJ	Tank	First	0x004F		

Experimentation with the different objects using the GTX tool is now left to the user as an exercise. The following tables are shown below as a reference. Consult the [ViSi Genie Reference Manual](#) for more information.

Command	Code	Parameter 1	Parameter 2	Parameter 3	Parameter 4	Parameter N	Checksum
READ_OBJ	0x00	Object ID	Object Index	-	-	-	Checksum
WRITE_OBJ	0x01	Object ID	Object Index	Value (msb)	Value(lsb)	-	Checksum
WRITE_STR	0x02	String Index	String Length	String (1 byte chars)		-	Checksum
WRITE_STRU	0x03	String Index	String Length	String (2 byte chars)		-	Checksum
WRITE_CONTRAST	0x04	Value	-	-	-	-	Checksum
REPORT_OBJ	0x05	Object ID	Object Index	Value (msb)	Value(lsb)	-	Checksum
REPORT_EVENT	0x07	Object ID	Object Index	Value (msb)	Value(lsb)	-	Checksum

This table is in section 2.1.2 of the [ViSi Genie Reference Manual](#) .

Object	ID	Input	Output	Notes
Dipswitch	0 (0x00)	✓	✓	
Knob	1 (0x01)	✓	✓	
Rockerswitch	2 (0x02)	✓	✓	
Rotaryswitch	3 (0x03)	✓	✓	
Slider	4 (0x04)	✓	✓	
Trackbar	5 (0x05)	✓	✓	
Winbutton	6 (0x06)	✓	✓	
Angularmeter	7 (0x07)		✓	
Coolgauge	8 (0x08)		✓	
Customdigits	9 (0x09)		✓	
Form	10 (0x0A)		✓	Used to set the current form
Gauge	11 (0x0B)		✓	
Image	12 (0x0C)			Displayed as part of form, no method to alter
Keyboard	13 (0x0D)	✓		Keyboard inputs are always single bytes and are unsolicited
Led	14 (0x0E)		✓	
Leddigits	15 (0x0F)		✓	
Meter	16 (0x10)		✓	
Strings	17 (0x11)		✓	
Thermometer	18 (0x12)		✓	
Userled	19 (0x13)		✓	
Video	20 (0x14)		✓	
Statictext	21 (0x15)			Displayed as part of form, no method to alter
Sound	22 (0x16)		✓	
Timer	23 (0x17)		✓	

This table is in section 3.3 of the [ViSi Genie Reference Manual](#) .

Proprietary Information

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

4D Systems 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 Systems products and services is continuous and published information may not be up to date. It is important to check the current position with 4D Systems.

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

Disclaimer of Warranties & Limitation of Liability

4D Systems makes no warranty, either expresses 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.

In no event shall 4D Systems 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 Systems, or the use or inability to use the same, even if 4D Systems has been advised of the possibility of such damages.

4D Systems 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 Systems and its suppliers specifically disclaim any expressed or implied warranty of fitness for High Risk Activities.

Use of 4D Systems' 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 Systems from any and all damages, claims, suits, or expenses resulting from such use. No licenses are conveyed, implicitly or otherwise, under any 4D Systems intellectual property rights.