4D LABS SEMICONDUCTORS

Workshop4

Integrated Development Environment

VISI-GENIE USER GUIDE

Document Revision: 2.2

Document Date: 27th February 2024

Table of Contents

1. Introduction to ViSi-Genie 5
2. Launch Workshop4 6
3. Create a New Project 6
3.1. Create a new 4D Systems Project 7
3.2. Create a new 4D Labs Project 9
4. Select ViSi-Genie 10
5. Using Serial with a Library 12
5.1. Area 1: Menus 13
5.2. Area 2: Ribbon with Icons 13
5.3. Area 3: List of Open Projects 13
5.4. Area 4: Form and WYSIWG Screen 14
5.5. Area 5: Object Inspector 14
5.6. Area 6: Message Window 15
6. A First ViSi-Genie Project 16
6.1. Adding Objects 16
6.2. Linking Objects 18
6.3. Controlling Multiple Objects 18
6.4. Chaining Objects 20
7. Objects 22
7.1. Buttons Object 25
7.1.1. Win Button 26
7.1.2. User Button 26
7.1.3. Animated Button 26
7.1.4. 4D Buttons 27
7.2. Digits Objects 28
7.2.1. LED Digits 28
7.2.2. Custom Digits 28
7.2.3. LED 28
7.2.4. User LED 28
7.3. Gauges Objects 29
7.3.1. Meter 29
7.3.2. Gauge 29
7.3.3. Angular Meter 30
7.3.4. Cool Gauge 30
7.3.5. Thermometer 30

7.3.6. Tank 31
7.3.7. Spectrum 31
7.3.8. Scope 32
7.3.9. Smart Gauge 32
7.4. Primitives Objects 33
7.4.1. Circle 33
7.4.2. Rectangle 33
7.4.3. Triangle 33
7.4.4. Line 34
7.4.5. Ellipse 34
7.4.6. Panel 34
7.5. Inputs Objects 35
7.5.1. Knob 35
7.5.2. Rotary Switch 35
7.5.3. Slider 35
7.5.4. Trackbar 36
7.5.5. Keyboard 36
7.5.6. DIP Switch 38
7.5.7. Rocker Switch 38
7.5.8. Color Picker 38
7.5.9. Smart Knob 39
7.5.10. Smart Slider 39
7.6. Labels Objects 40
7.6.1. Label 40
7.6.2. Static Text 40
7.6.3. Strings 40
7.7. System/Media Objects 42
7.7.1. Image 42
7.7.2. Video 43
7.7.3. Form 43
7.7.4. Sounds 44
7.7.5. Timer 45
7.7.6. User Images 45
7.8. I/O 46
7.8.1. Pin Input 46
7.8.2. Pin Output 46
7.9. Backgrounds 47
7.9.1. Border 47
7.9.2. Gradient 47
7.9.3. Scale 47

7.10. Magic Objects 48
7.10.1. Magic Event 48
7.10.2. Magic Touch 48
7.10.3. Magic Move 48
7.10.4. Magic Release 49
7.10.5. Magic Keyboard and Color Picker Event 49
7.10.6. Magic Code 49
7.10.7. Magic Object 49
7.11. Selection Tool 50
8. ViSi-Genie Communications Protocols 51
8.1. Genie Standard Protocol 51
8.1.1. Protocol Definitions 51
8.1.2. Command and Parameters Table 52
8.1.3. Command Set Messages 53
8.1.4. Acknowledgement Bytes Table 53
8.1.5. Genie Advanced Protocol 53
8.2. Object Types Table 54
9. Integrated Debugger 55
10. Communication Terminal 57
11. Application Notes 58
12. Revision History 59
13. Legal Notice 60
14. Contact Information 60

1. Introduction to ViSi-Genie

This user guide provides an introduction to ViSi-Genie, the codeless rapid development tool for designing and building graphic user interface on 4D Labs processor-based displays.

ViSi-Genie is a breakthrough in the way 4D Labs' graphic display modules are programmed, it provides an easy method for designing complex Graphics User Interface applications without any coding. It is an environment like no other, a code-less programming environment that provides the user with a rapid visual experience, enabling a simple GUI application to be 'designed' from scratch in literally seconds.

ViSi-Genie does all the background coding, no 4DGL programming language to learn, it does it all for you.

Pick and choose the relevant objects to place on the display, much like the ViSi environment, yet without having to write a single line of code. The full animation of the objects is done under-the-hood, such as pressing a button or moving the thumb of the slider. Each object has parameters which can be set, and configurable events to animate and drive other objects or communicate with an external host.

Simply place an object on the screen, position and size it to suit, set the parameters such as colour, range, text, and finally select the event you wish the object to be associated with, it is that simple. Objects are classified in three different groups:

INPUT OBJECTS, as a button or a keyboard, **OUTPUT OBJECTS**, as a gauge or a meter, and **COMBINED OBJECTS** or **INPUT/OUTPUT OBJECTS**, as a slider which acts as both an input and an output.

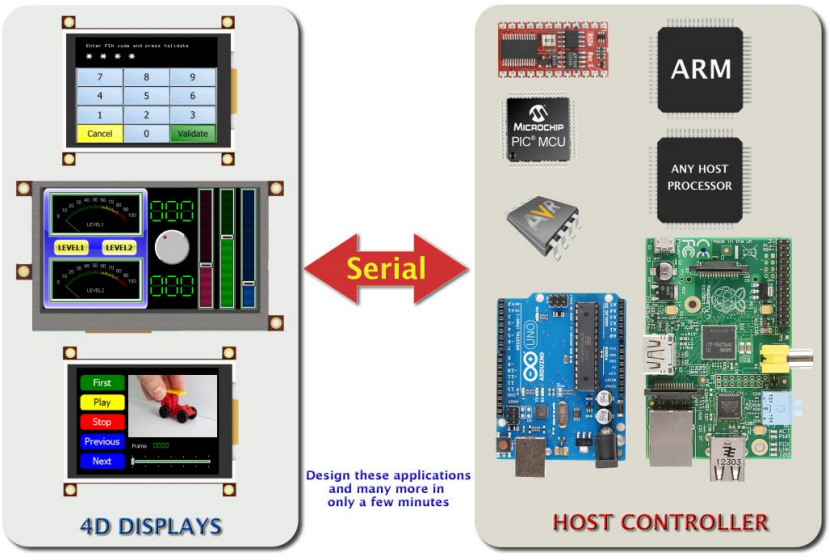
In seconds you can transform a blank display into a fully animated GUI with moving meters, animated press and release buttons, and much more. **All without writing a single line of code!**

ViSi-Genie provides the user with a feature rich rapid development environment, second to none.

This document should be used in conjunction with the *ViSi-Genie Reference Manual*.

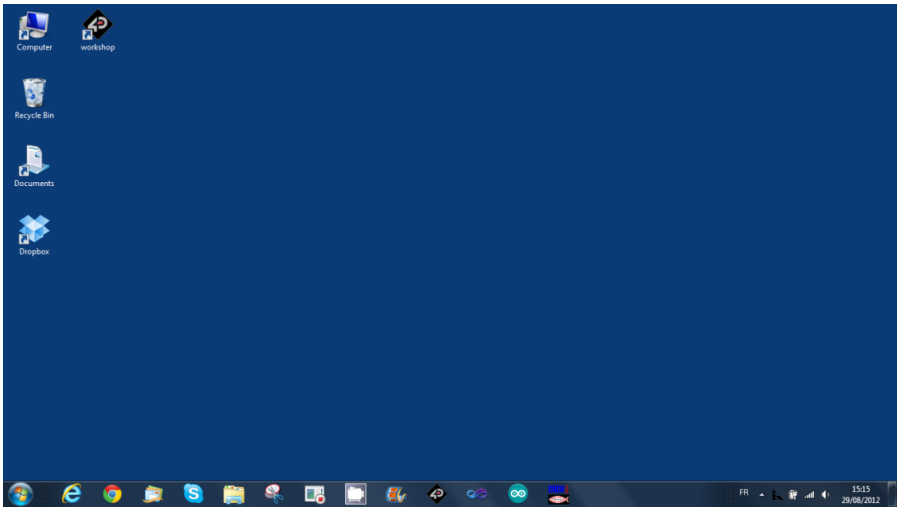
ViSi-Genie is included in the integrated development environment Workshop4. To install Workshop4, please refer to the document *Workshop4 IDE Installation Guide*.

Note: ViSi-Genie is not available on the Goldelox platform.



2. Launch Workshop4

There is an alias for Workshop4 on the desktop:



Launch 4D Workshop by double-clicking on the icon:

3. Create a New Project

Workshop4 opens and displays the **Recent** page:

On the Recent page, there are three options:

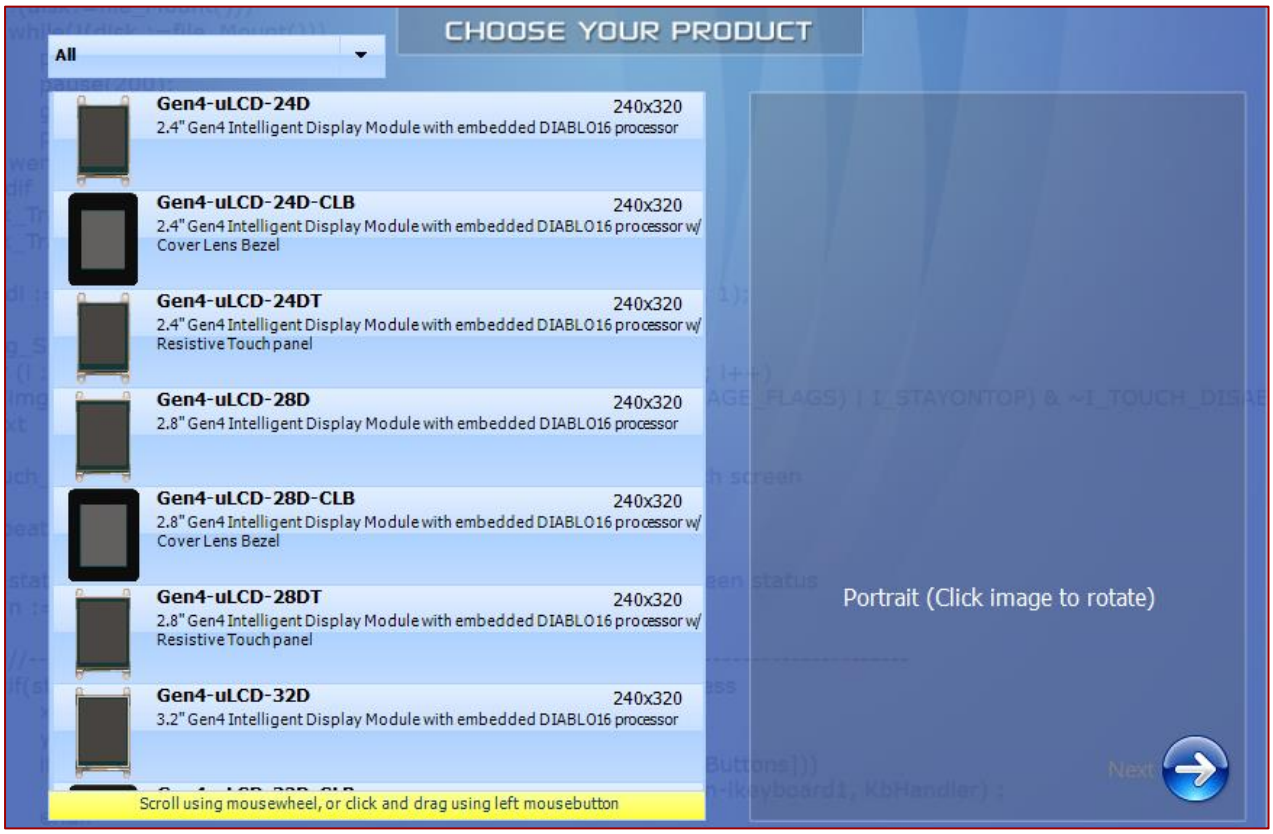
1. **Create a new 4D Systems Project:** for creating projects using 4D Systems display modules
2. **Create a new 4D Labs Project:** for creating projects using a 4D Labs processor with a custom LCD
3. **Create a new Project:** for creating a project based on the last used settings

Note: As of writing, option 2 (Create a new 4D Labs Project) is not yet available and it will be released soon.

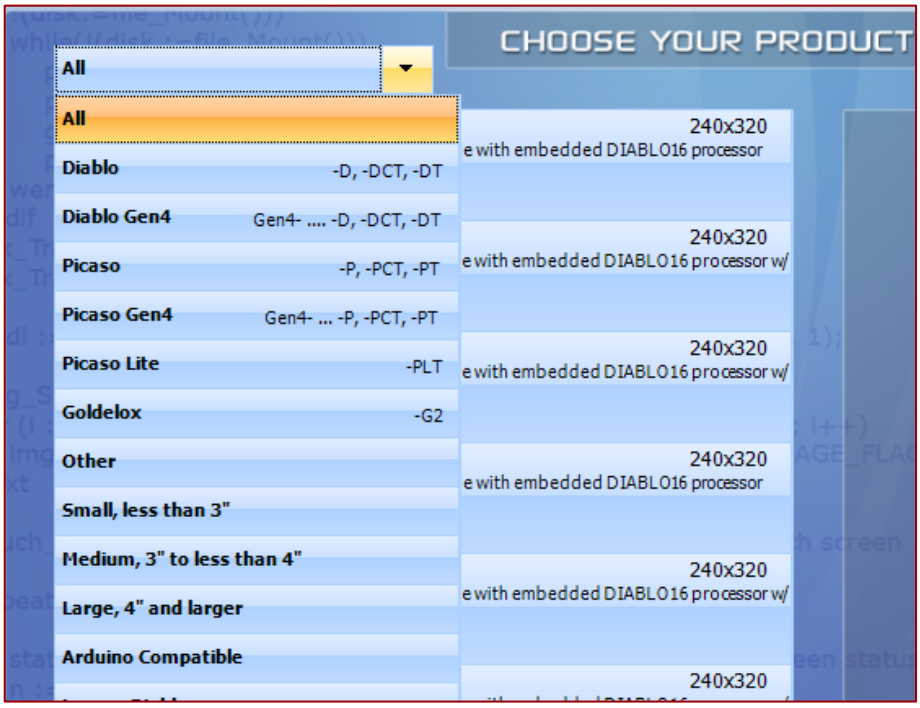
3.1. Create a new 4D Systems Project

To create a new project using a 4D Systems display module, click on the 4D Systems icon, as shown below.

The Choose-your-Product window appears and the available 4D Systems display modules are shown.



The dropdown menu at the top can be used to filter the products by categories.



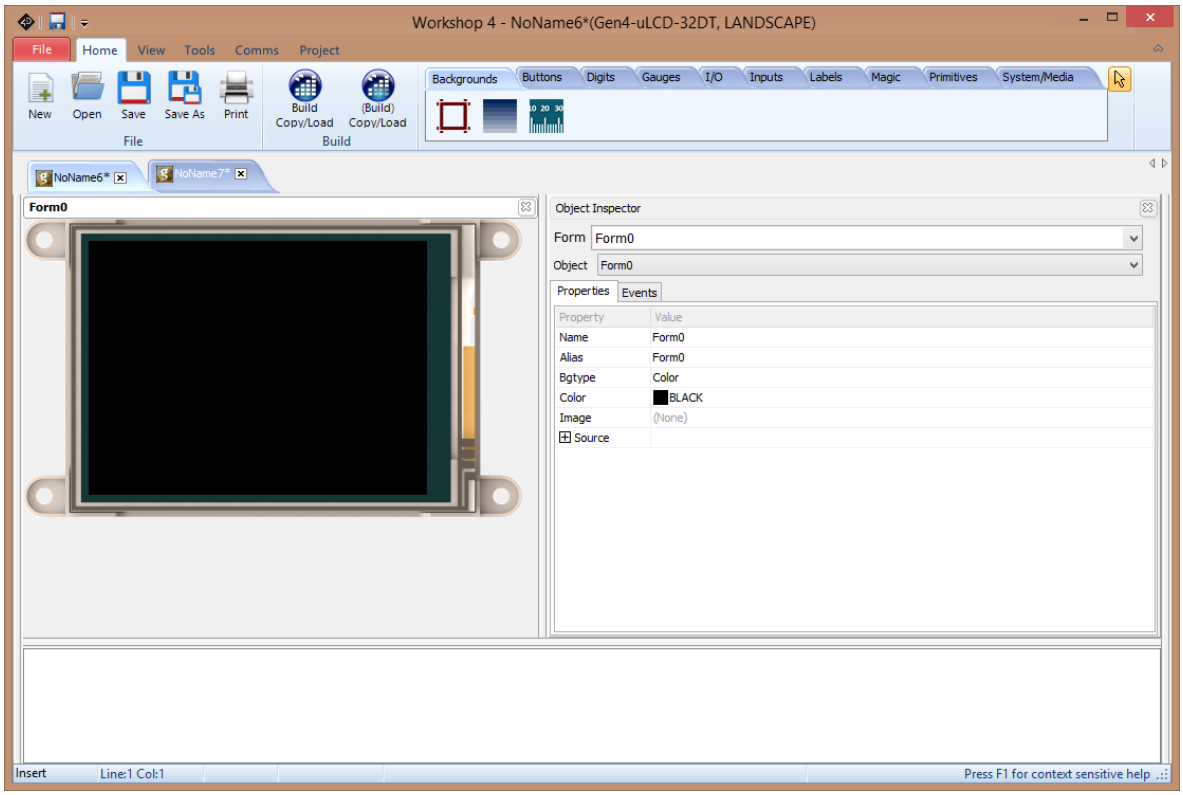
3.2. Create a new 4D Labs Project

As of writing, this option is not yet available, and it will be released soon.

4. Select ViSi-Genie

The main window now asks for the environment to be used. Select ViSi-Genie.

The development environment is now displayed:



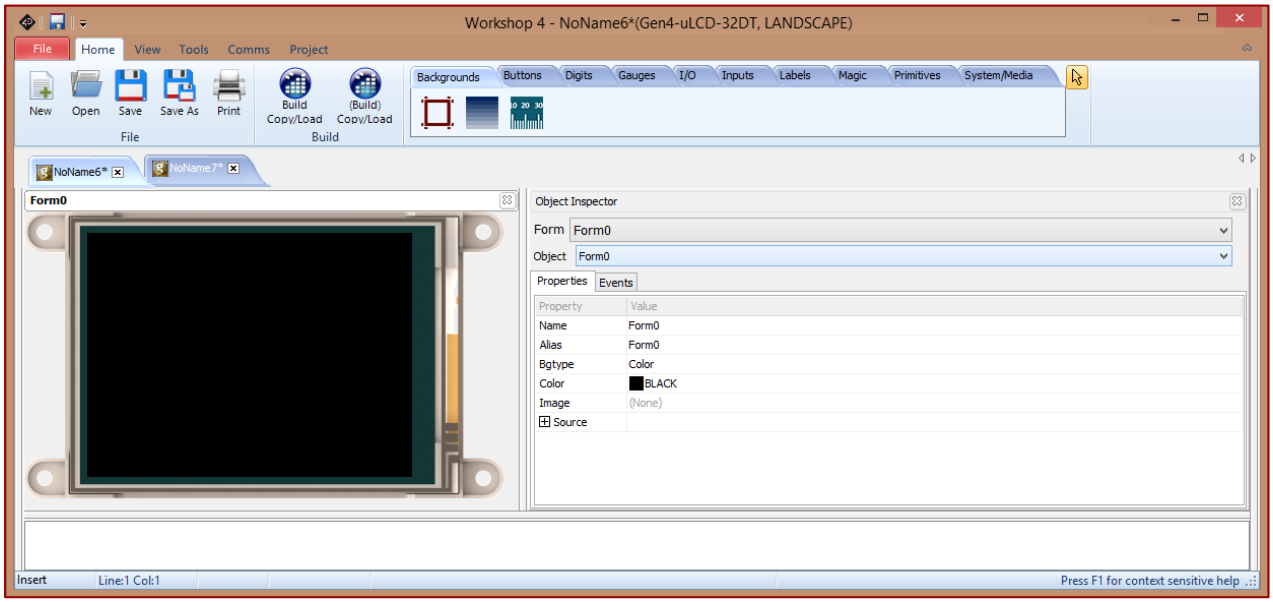
Workshop4 displays an empty screen, called Form0.

- A **project** consists of one or more forms.
- A **form** is like a page on the screen.
- The form includes **objects**, like sliders, displays or keyboards.

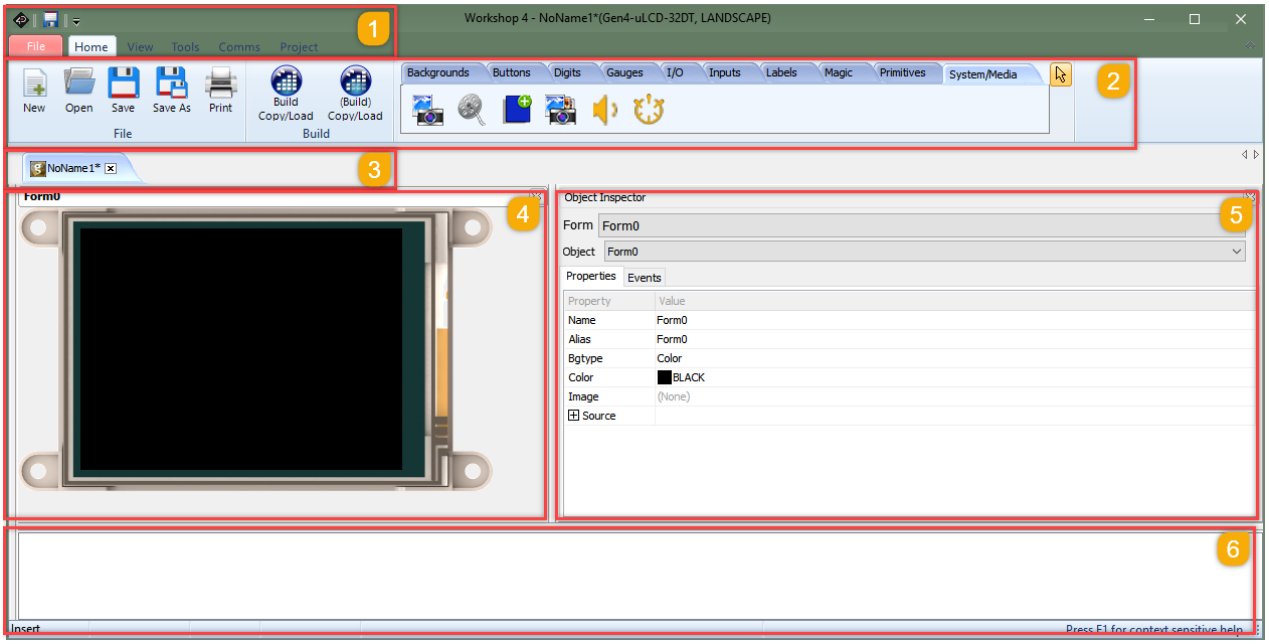
You are ready to start.

5. Using Serial with a Library

The main screen appears.



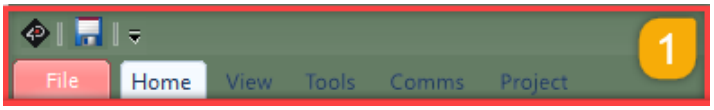
Let's discuss the different areas. There are six different areas, from left to right, for top to bottom:



1. Menus;
2. Ribbon with icons;
3. List of open projects;
4. Form and WYSIWYG screen where to place the objects;
5. Object inspector, where properties and events are defined;
6. Messages about errors, warnings and notices.

5.1. Area 1: Menus

The menus include standard Windows options. Each menu displays a specific ribbon.



The debugger called **Genie Test Executor** is located under the Tool menu.

5.2. Area 2: Ribbon with Icons

For the Home menu, the ribbon includes the file related buttons and the objects grouped in panes:

The icons related to the files include **New** project, **Open** project, **Save** project, **Save as** project, **Print** project, and **Build** project.

The objects are grouped in ten panes, with input objects, output objects and composite objects. Just click on an object to select it.

5.3. Area 3: List of Open Projects

On top of the What-You-See-Is-What-You-Get (WYSIWYG) screen, the open projects are displayed:

Click on the tab to open it or on the cross to close it.

5.4. Area 4: Form and WYSIWG Screen

The form represents a WYSIWYG screen.

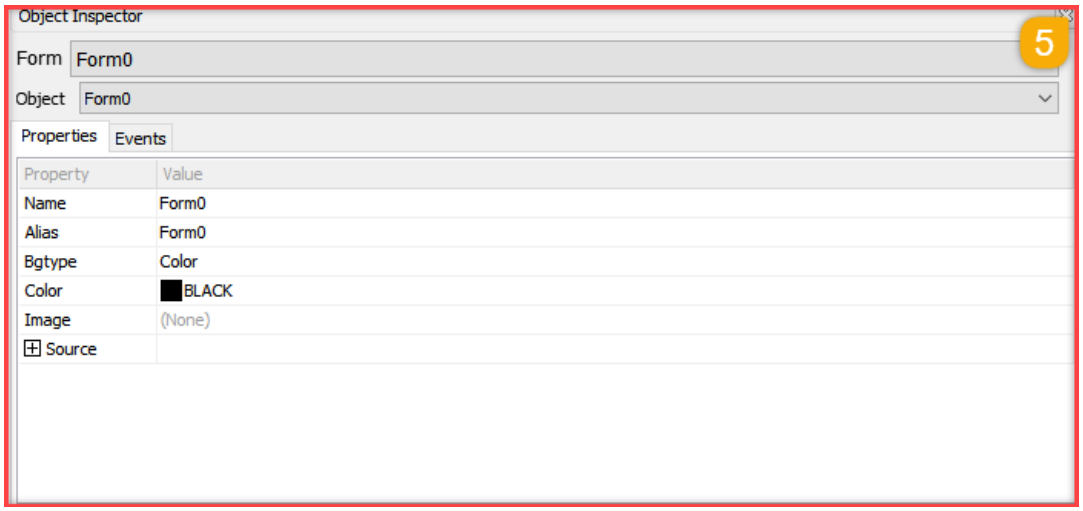
The active form is displayed there, with its objects. Objects are picked from the panes and can be resized and moved.

Click on an object to select it.

5.5. Area 5: Object Inspector

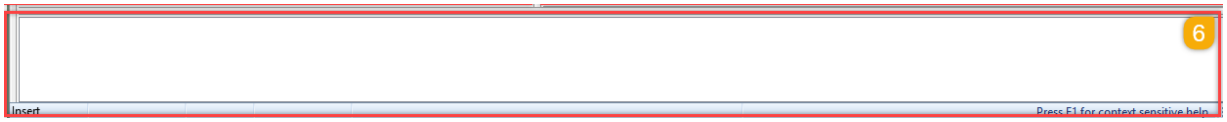
The object inspector provides all the information on the selected object:

- properties, as size and position;
- and events, where actions are defined.



5.6. Area 6: Message Window

The message window displays errors, warnings and notices after the project is built.



Before starting using the Workshop4, we need to connect the screen and prepare a micro-SD card.

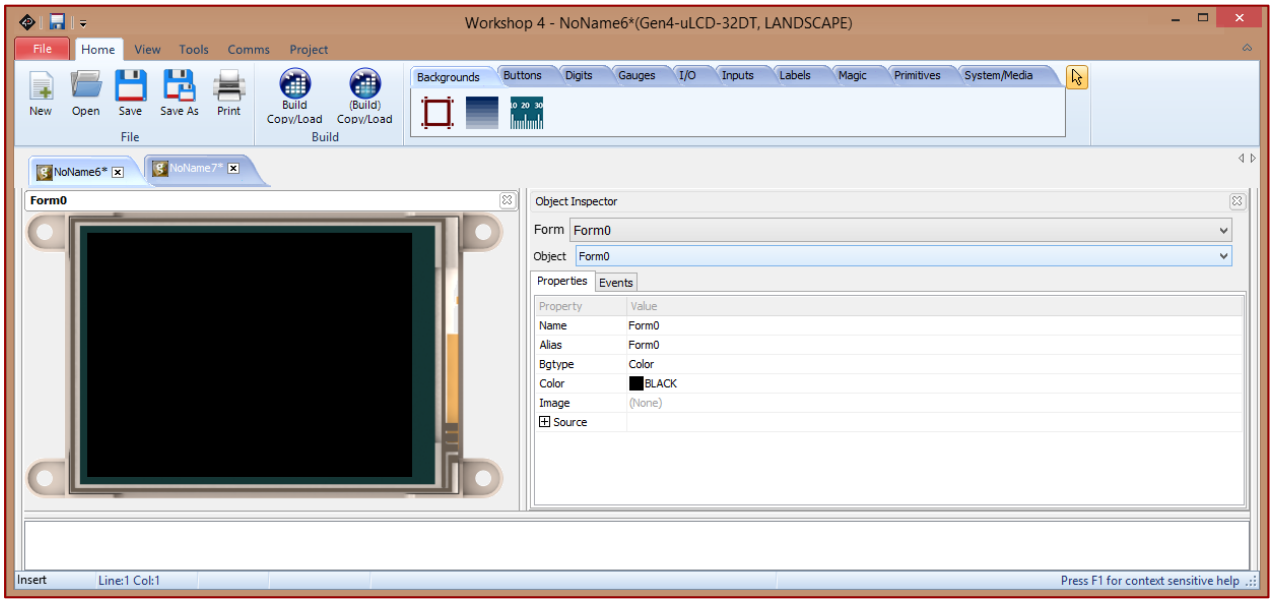
Note: For more information about connecting the screen, please refer to the Workshop4 User Guide.

The micro-SD card shall be FAT16-formatted. Partition can't exceed 4 GB.

Note: For more information about formatting the micro-SD card, please check the details on chapter *MicroSD Card Format* described in Workshop4 User Guide.

6. A First ViSi-Genie Project

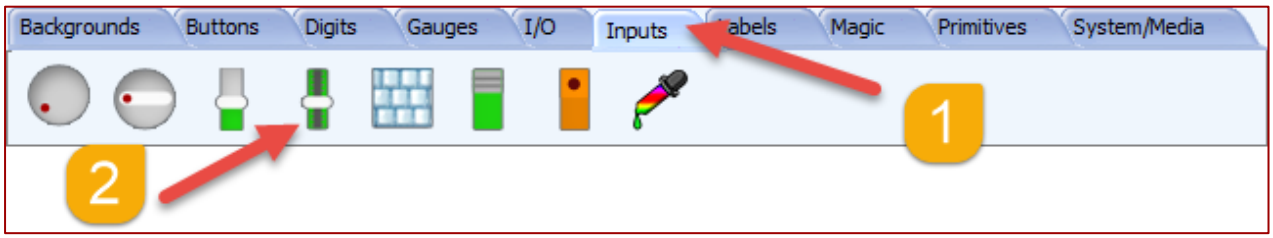
Workshop4 displays an empty screen, called **Form0**.
A **form** is like a page on the screen. The form includes **objects**, like sliders, displays or keyboards. A **project** consists of one or more forms.



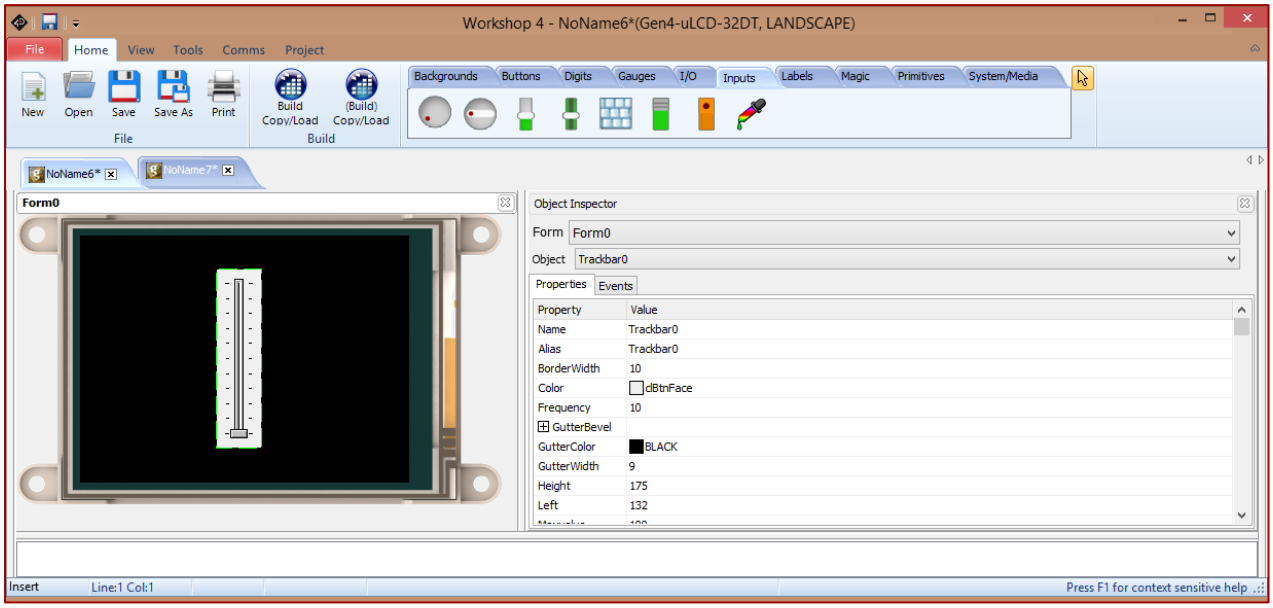
The form is empty.
We are going to build a form with two objects: a track bar that updates a meter.

6.1. Adding Objects

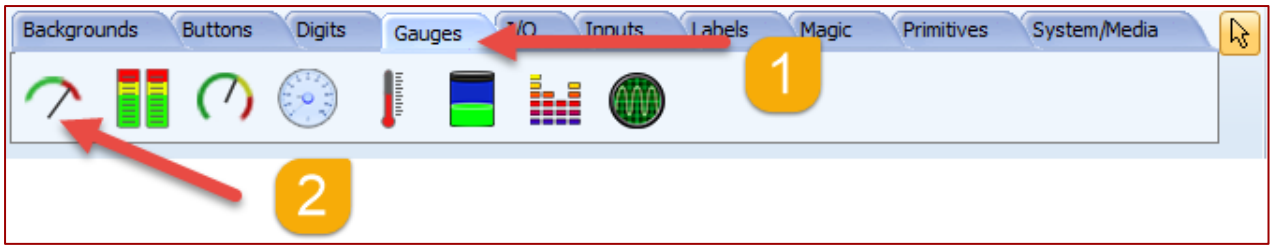
The track-bar is an **input object** and the meter is an **output object**.
Select the **Inputs** pane then the Trackbar object.



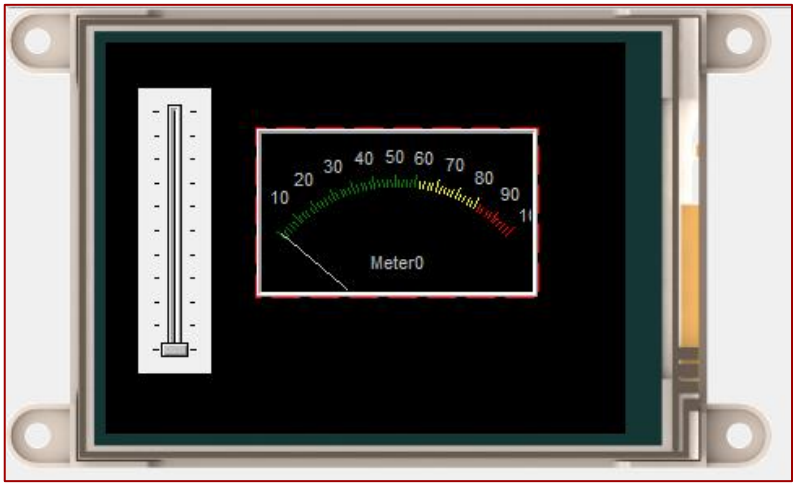
...and then click on the desired location on the form to place it:



Now, the same applies for the meter. Select the **Gauges** pane then the Meter object.



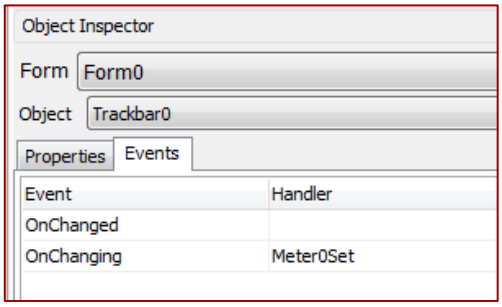
The final form looks like:



Note: For a step-by-step example of a project, please refer to the application note ViSi-Genie: Getting Started with PICASO Displays or ViSi-Genie: Getting Started with DIABLO-16 Displays.

6.2. Linking Objects

Now, the objects need to be linked: moving the track-bar updates the meter.
Moving the track-bar raises an **event**, called **OnChanging**. When an **OnChanging** event arises, a message is sent to the meter with the value.

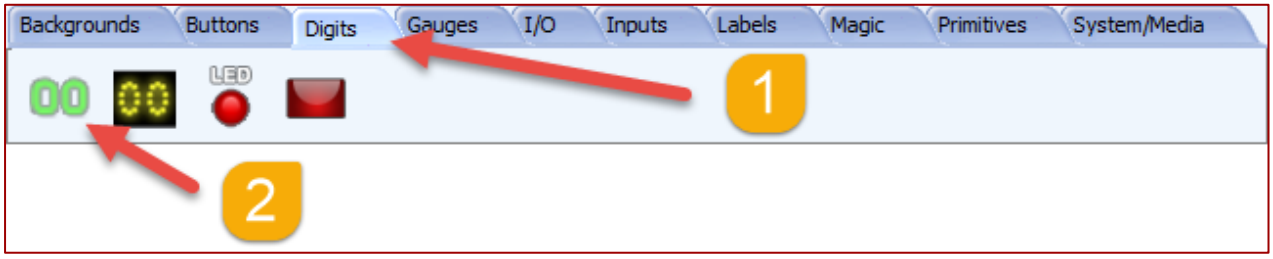


For the end-user, each time he moves the trackbar, the meter is updated accordingly.

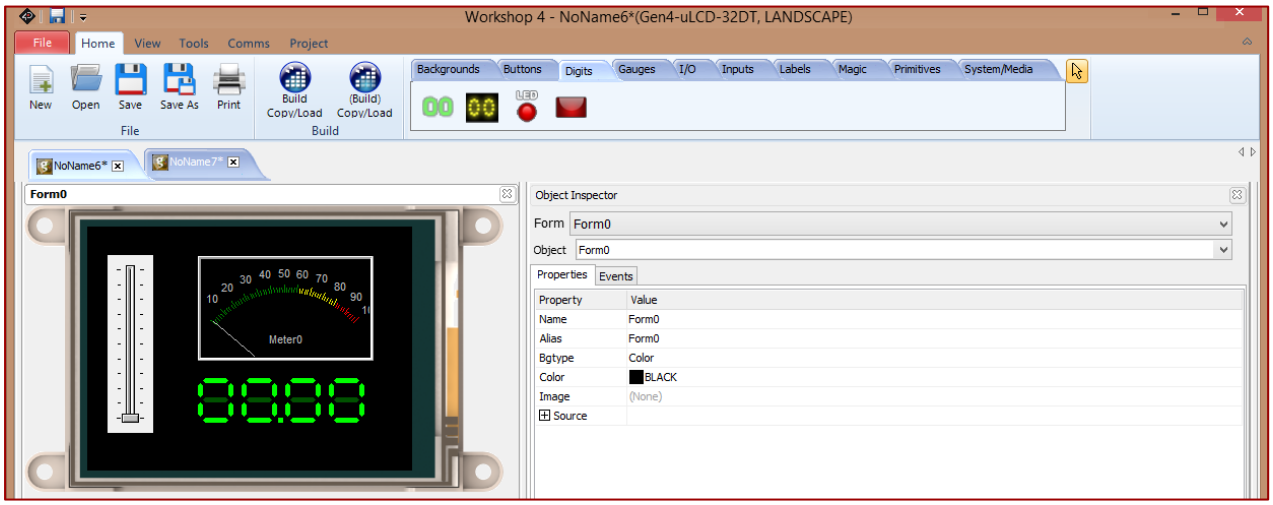
Note: For a detailed presentation of the onChanging and onChanged events, please refer to the application note ViSi-Genie: onChanging and onChanged Events.

6.3. Controlling Multiple Objects

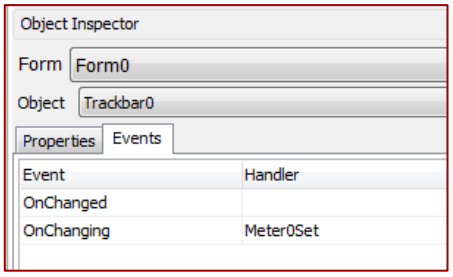
As described in the previous section, an object sends a message to another single object.
Select the **Digits** pane then the LedDigits object.



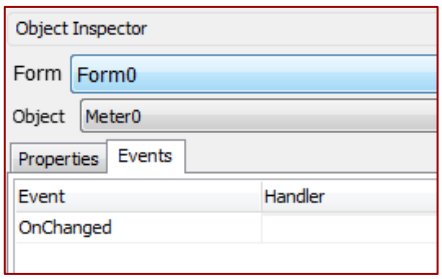
...and place it on the form. The final form looks like:



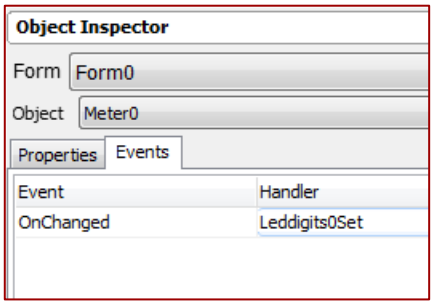
As previously, moving the track-bar raises the **OnChanging** event, which sends a message to **Meter0** with the value.



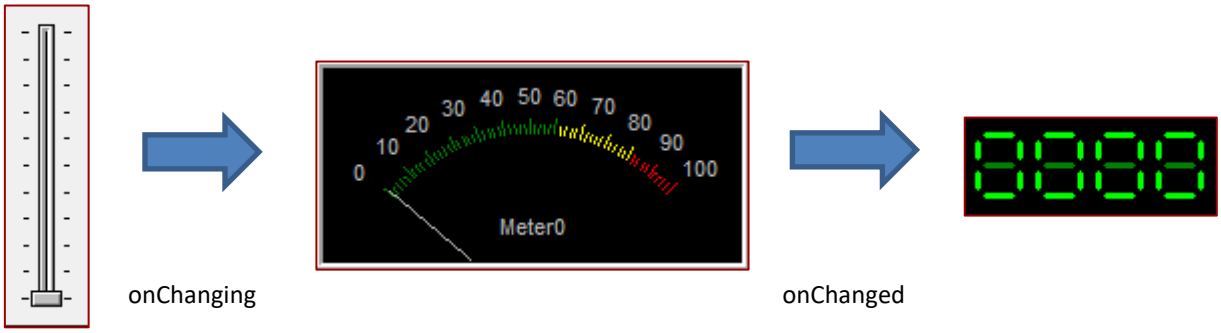
Now, the meter has the event **OnChanged** raised when the meter receives a new value.



An action can be associated to that event to send the value to the **LedDigits0** object:



Summarising:



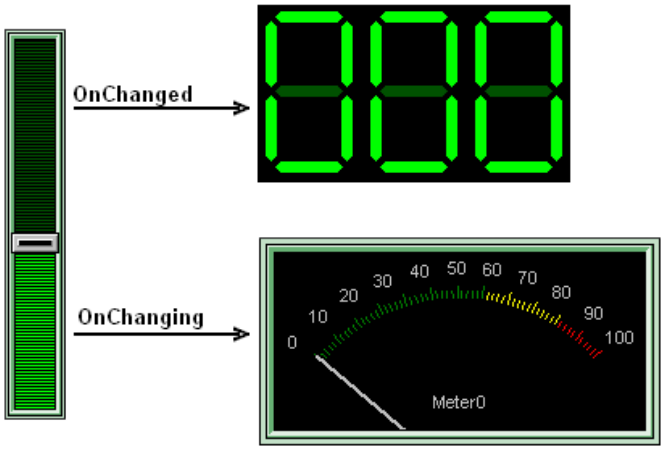
onChanging onChanged

- Moving the track-bar raises the **OnChanging** event, which sends a message to **Meter0** with the value;
- The meter **Meter0** displays the new value and raises the **OnChanged** event, which sends a message to **LedDigits0** with the value.

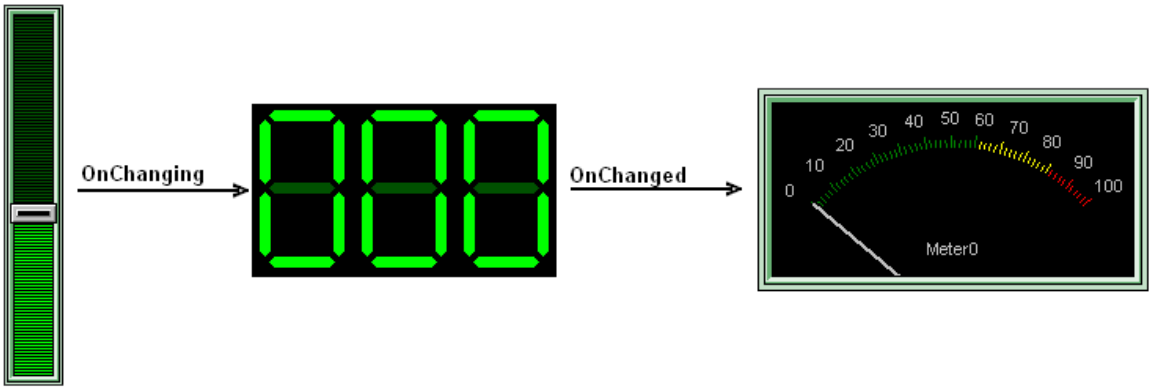
That way, multiple objects can be controlled.

6.4. Chaining Objects

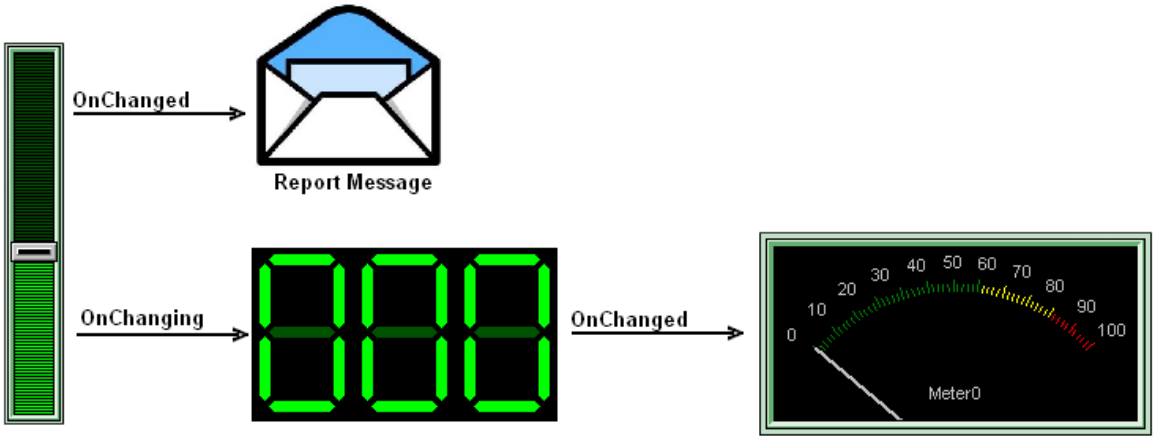
Combining the **OnChanged** and **OnChanging** events with sending messages from one object to another allows multiple configurations:



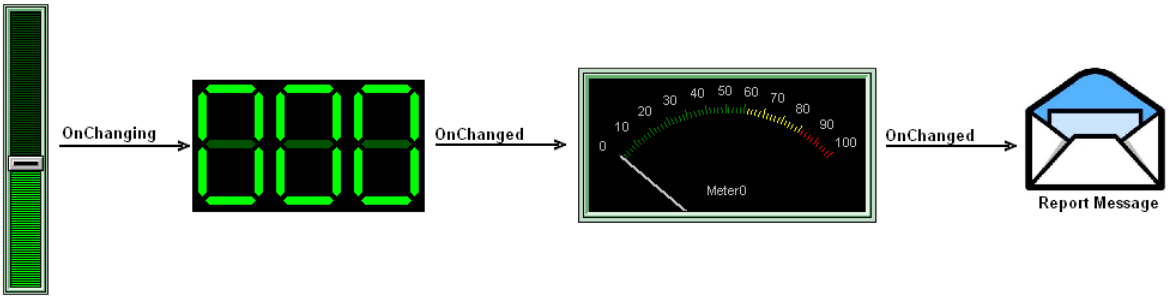
Another configuration with the same result:



A message is sent to the host controller once the track-bar has been released:



Another configuration with the same result:



Note: For more information on the interfacing of ViSi-Genie with a host micro-controller, please refer to the 4D Systems application note ViSi-Genie Connecting a 4D Display to an Arduino Host.

7. Objects

ViSi-Genie relies on three groups of objects:

- The **INPUT OBJECTS** produce stimuli data for INPUT type objects or directly to Serial output. The animation for these objects is done under the hood, for example the slider thumb movements, etc. A button press can launch a sub-form or can send out serial data or cause another event to occur.
  Example: a button.
- The **OUTPUT OBJECTS** only react to OUTPUT stimuli. The stimulus data can come from the Serial port or an INPUT object. They produce no input data or stimuli. The animation for these objects is performed under the hood, for example incoming serial data can move the needle of the meter, etc. OUTPUTs can be set regardless of whether they are displayed on the current form, when the form containing them is displayed, they are displayed with their current value.
  Example: a meter.
- Actually, most objects are **COMBINED OJECTS** or **INPUT/OUTPUT OBJECTS**. Most input objects can also function as outputs, with the notable exception of Keyboards. Certain objects need both an input stimuli as well as produce an output event. For example, a slider thumb position may need to be remotely controlled from incoming serial data. A button may need to be animated not only using the touch screen but via serial data.
  Example: a slider.

Worskhop4 PRO has an extension called “Genie Magic”. Under Genie Magic there are additional objects that allow the user to add 4DGL code at various points in the program. These objects are called “Magic Objects”. These are found under the Magic pane.

Note: The Magic objects are available only in Workshop4 PRO.

Here is the summary of the input, output and combined objects. A combined object is ticked both as input and output.

SUMMARY OF OBJECTS			
Pane	**Object**	**Input**	**Output**
Button	Win Button	✓	✓
	4D Button	✓	✓
	Ani Button	✓	✓
	User Button	✓	✓
Digits	Led Digits		✓
	Custom Digits		✓
	Led		✓
	User Led		✓
Gauges	Meter		✓
	Gauge		✓
	Angular Meter		✓
	Cool Gauge		✓
	Thermometer		✓
	Spectrum		✓
	Scope		✓
	Tank		✓
	Smart Gauge		✓
Primitives	Circle		
	Rectangle		
	Triangle		
	Line		
	Ellipse		
	Panel		

SUMMARY OF OBJECTS			
Pane	**Object**	**Input**	**Output**
Inputs	Knob	✓	✓
	Rotary Switch	✓	✓
	Slider	✓	✓
	Track Bar	✓	✓
	Keyboard	✓	
	Dip Switch	✓	✓
	Rocker Switch	✓	✓
	Color Picker	✓	✓
	Smart Slider	✓	✓
	Smart Knob	✓	✓
Labels	Label		
	Static Text		
	Inputs	Knob	✓

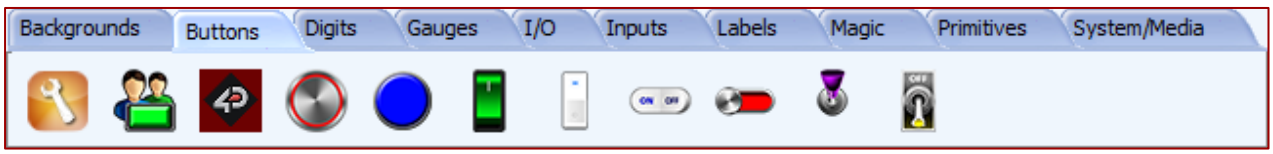
SUMMARY OF OBJECTS			
Pane	**Object**	**Input**	**Output**
System / Media	Image		
	Video		✓
	Form		✓
	Sound		✓
	Timer		✓
	User Images		✓
I/O	Pin Input	✓	
	Pin Output		✓
Backgrounds	Border		
	Gradient		
	Scale		
Genie-Magic (available in Worshop4 PRO)	Event		
	Touch		
	Move		
	Release		
	Keyboard + Colorpicker		
	Magic Code		
	Magic Object		

Each object is presented with its button on the left and an example on the right when used on a form.

There is no Z-order in ViSi-Genie. Objects are always drawn on the display 'background'. The display background is the Form background (colour or image) with the 'Backgrounds' objects 'added'.

For objects that are transparent, or have a transparent aspect (eg corners of WinButtons) this means what shows through the transparent region will be the Form background + 'Backgrounds', not what may appear underneath when viewing in Workshop. Any visual effects that need to be 'underneath' objects should be included in a form background image.

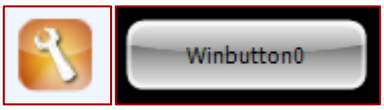
7.1. Buttons Object



The Buttons pane contains the WinButton, User Button, Animated Button, and the 4D Buttons. These objects have one single event, onChanged. Buttons can be linked together to form a group through a matrix. When one button of the matrix is pressed, the previous one is released. Buttons can also be momentary or toggle type.

Note: For more information on the button objects, please refer to the application notes as they become available.

7.1.1. Win Button



This object has one single event, onChanged.

Note: For more information on the win button object, please refer to the 4D Systems application note ViSi-Genie: Advanced Buttons.

7.1.2. User Button

A generic button object for the users to create their own buttons with. The user button has four states – up, up pressed, down, and down pressed. The user provides the image for each of these states. The user button can be turned in to either a momentary, toggle, or matrix type.

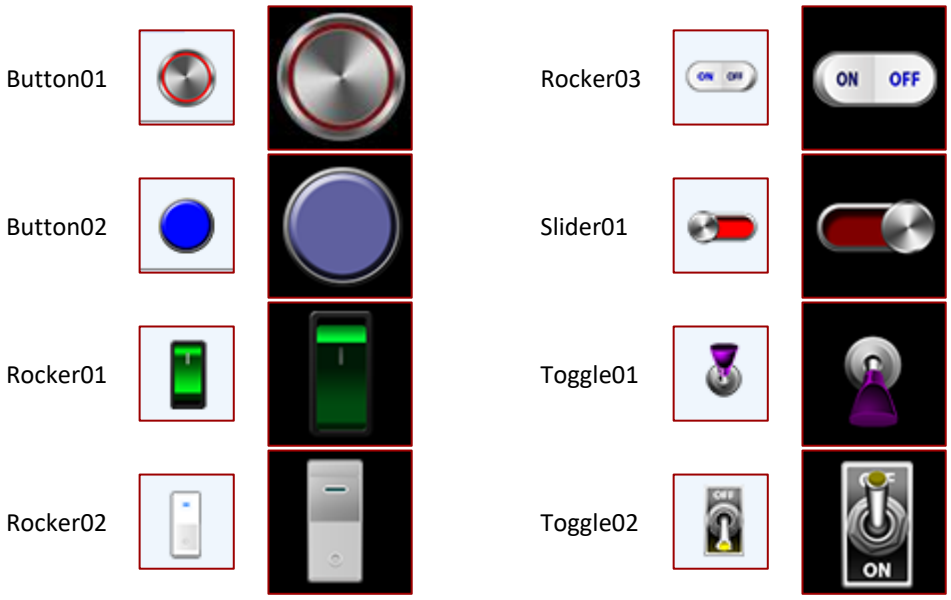
Note: For more information on the user button object, please refer to the 4D Systems application note ViSi-Genie User Button.

7.1.3. Animated Button

The animated button plays a sequence of images when touched. The user provides these images and sets the delay interval with which they are displayed. The animated button can be turned in to either a momentary, toggle, or matrix type.

Note: For more information on the animated button object, please refer to the 4D Systems application note ViSi-Genie Animated Button.

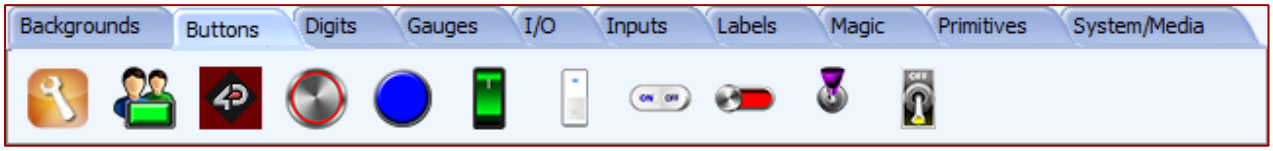
7.1.4. 4D Buttons



There are various 4D button types available in Workshop. The user can choose among the predefined sizes and styles. The 4D buttons can also be turned in to either a momentary, toggle, or matrix type.

Note: For more information on the 4D button objects, please refer to the 4D Systems application note ViSi-Genie: 4D Buttons.

7.2. Digits Objects

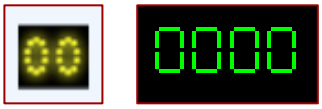


The Digits pane contains 4 different displays.

7.2.1. LED Digits

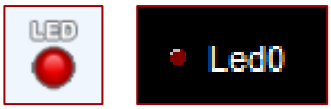
The number of digits, the decimal place, the size, and the leading zeros can be customised.
This object has one single event, onChanged, very useful to send the value received.

7.2.2. Custom Digits



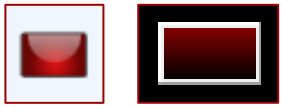
This object offers no customisation.
This object has one single event, onChanged, very useful to send the value received.

7.2.3. LED



The size, the label, the font and the colour can be customised.
This object has one single event, onChanged, very useful to send the value received.

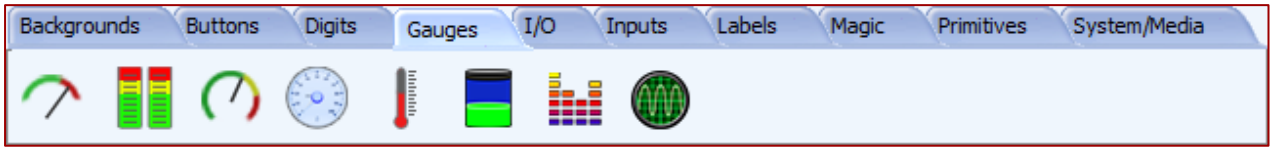
7.2.4. User LED



The size and the colour can be customised.
This object has one single event, onChanged, very useful to send the value received.

Note: For more information on the Digits objects, please refer to the 4D Systems application note ViSi-Genie Digital Displays.

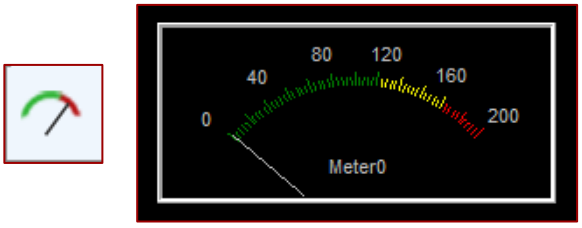
7.3. Gauges Objects



The Gauges pane contains 8 specialised displays.

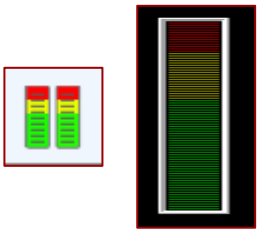
Note: For more information on the Gauge objects, please refer to the 4D Systems application note ViSi-Genie: Gauges. Dedicated application notes for some of the gauge objects are also available.

7.3.1. Meter



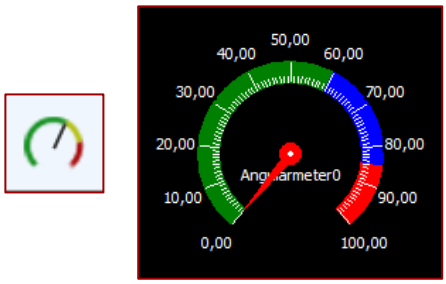
The meter displays a value in a dial. Minimum and maximum, number of intervals and scales, all colours are fully configurable, among other options. The meter is an output object and can send a message when changed. This object has one single event, onChanged, very useful to send the value received.

7.3.2. Gauge



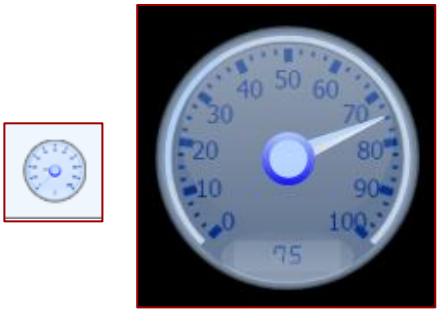
The gauge displays a value in a dial. Minimum and maximum, number of intervals, scales and three palettes, all colours are fully configurable, among other options. The gauge is an output object and can send a message when changed. This object has one single event, onChanged, very useful to send the value received.

7.3.3. Angular Meter



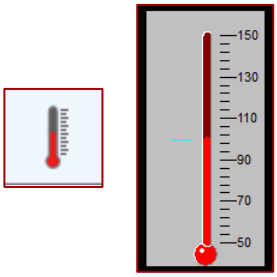
The angular meter displays a value in a dial. Minimum and maximum, number of intervals, scales and three zones, all colours are fully configurable, among other options. The angular meter is an output object and can send a message when changed. This object has one single event, onChanged, very useful to send the value received.

7.3.4. Cool Gauge



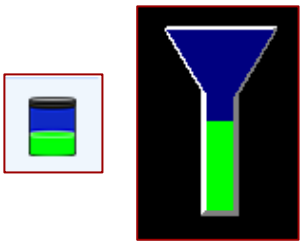
The cool gauge displays a value in a dial. Minimum and maximum, linear or logarithmic scales, all colours are fully configurable, among other options. The cool gauge is an output object and can send a message when changed. This object has one single event, onChanged, very useful to send the value received.

7.3.5. Thermometer



This object offers no customisation. This object has one single event, onChanged, very useful to send the value received.

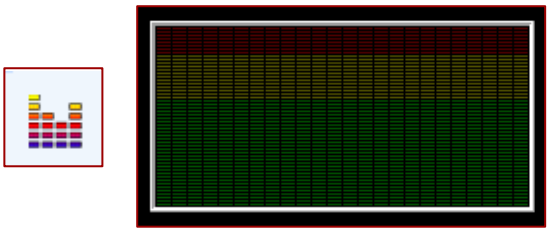
7.3.6. Tank



The tank object can be used to visually represent how much of an area is occupied. Minimum and maximum values and all colours are fully configurable, among other options. The tank is an output object and can send a message when changed. This object has one single event, onChanged, very useful to send the value received.

Note: For more information on the tank object, please refer to the 4D Systems application note ViSi-Genie Tank.

7.3.7. Spectrum

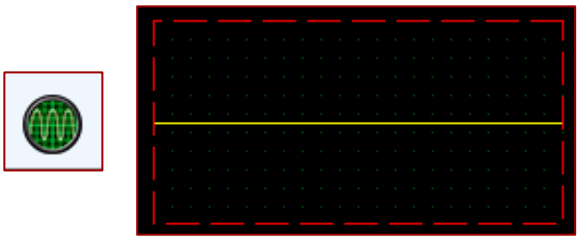


The spectrum can be used to visually indicate multiple quantity levels. It can be used as a bar graph or as an audio equalizer display, among other possible applications. The spectrum object is designed to be controlled or written to by an external host controller. Values can only be meaningfully be written to a Spectrum whilst its form is displayed. Each value written to a spectrum is comprised of two bytes, the first byte is the bar (0 to Columns-1), the second byte is the value (0 to maximum value). If the form on which the Spectrum appears is changed all displayed values should be considered lost and must be resent from the host when the form containing the spectrum is redisplayed.

The spacing between, number, and width of bars and area colours are all configurable, among other options. This object has no event.

Note: For more information on the spectrum object, please refer to the 4D Systems application note ViSi-Genie Spectrum.

7.3.8. Scope



The scope can display a maximum of four signal traces on the screen. It is designed to be driven by an external host controller. The number and colour of the traces, the rate at which the scope is updated, the amplification/attenuation of the trace in the Y direction, the compression/expansion of the trace in the X direction, and the properties of the graticule are all configurable, among other options. This object has no event.

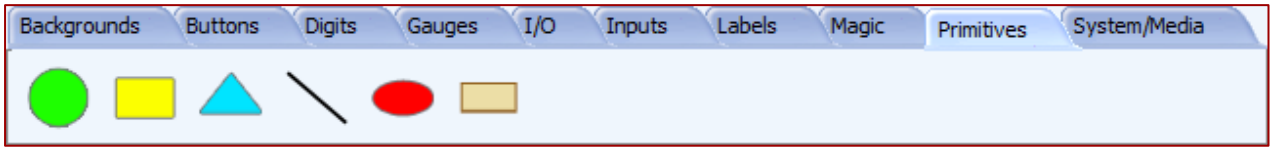
Note: For more information on the scope object, please refer to the 4D Systems application note ViSi-Genie Single Trace Scope.

7.3.9. Smart Gauge

The smart gauge is a highly customizable object that is accessible for **PRO** version users. This object allows user to create custom output objects using multiple image frames. Each layer, except the face image, can be manipulated to move horizontally, vertically or rotate with respect to a specified point. It is also possible to use different static images and show them in order in a single layer. The smart gauge is an output object and can send a message when changed. This object has one event, onChanged, very useful to send the value received.

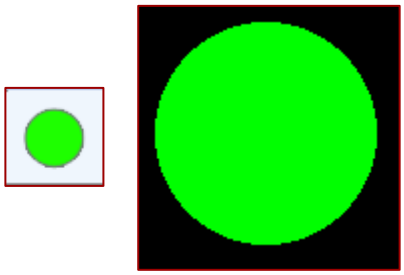
Note: For more information on the Smart Gauge object, please refer to the Smart Widgets Editor Manual.

7.4. Primitives Objects



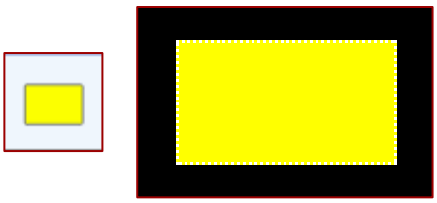
The Primitives pane offers standard static drawings.

7.4.1. Circle



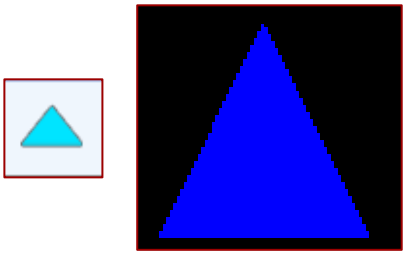
The colour and the option of empty or solid can be customised.
This object has no event.

7.4.2. Rectangle



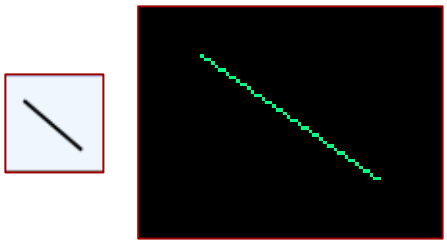
The colour, the outline and the option of empty or solid can be customised.
This object has no event.

7.4.3. Triangle



The colour, the outline and the option of empty or solid can be customised.
This object has no event.

7.4.4. Line

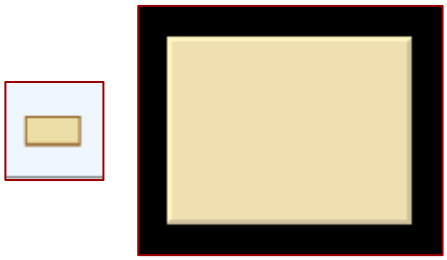


The colour and the pattern can be customised.
This object has no event.

7.4.5. Ellipse

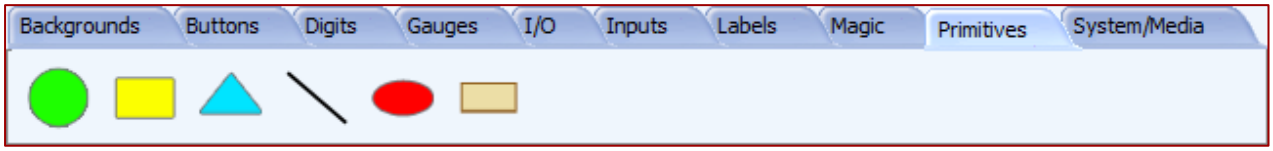
The colour and the option of empty or solid can be customised.
This object has no event.

7.4.6. Panel



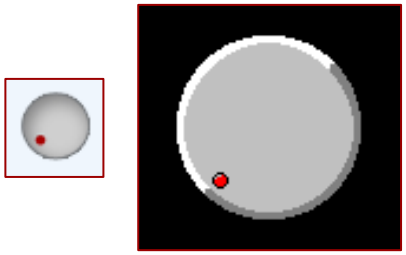
The colour, the outline, the state lowered or raised can be customised.
This object has no event.

7.5. Inputs Objects



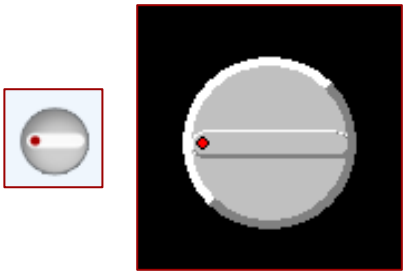
The Inputs pane contains rotary selectors, linear selectors, keyboards, switches, and color picker.

7.5.1. Knob



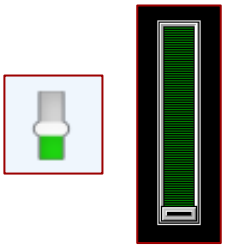
The minimum and maximum angles, the back and the handle can be customised.
This object has two events, **onChanged** and **onChanging**.

7.5.2. Rotary Switch



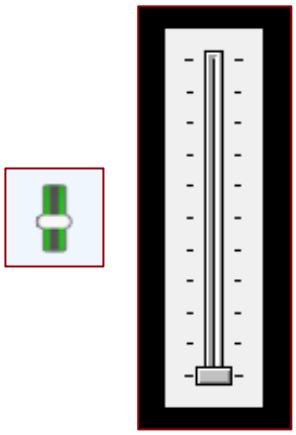
The minimum and maximum angles, the positions and labels, the switch and the winch colours can be customised.
This object has two events, **onChanged** and **onChanging**.

7.5.3. Slider



The minimum and maximum values, the vertical or horizontal orientations, the colours can be customised.
This object has two events, **onChanged** and **onChanging**.

7.5.4. Trackbar



The minimum and maximum values, the vertical or horizontal orientations, the frequency and ticks, the colours can be customised.

This object has two events, **onChanged** and **onChanging**.

7.5.5. Keyboard

ViSi-Genie comes with various defined keyboards:

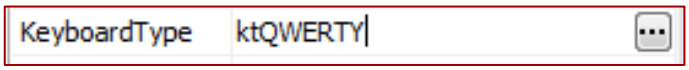
- QWERTY keyboard, by default,

- Cell-phone keyboard

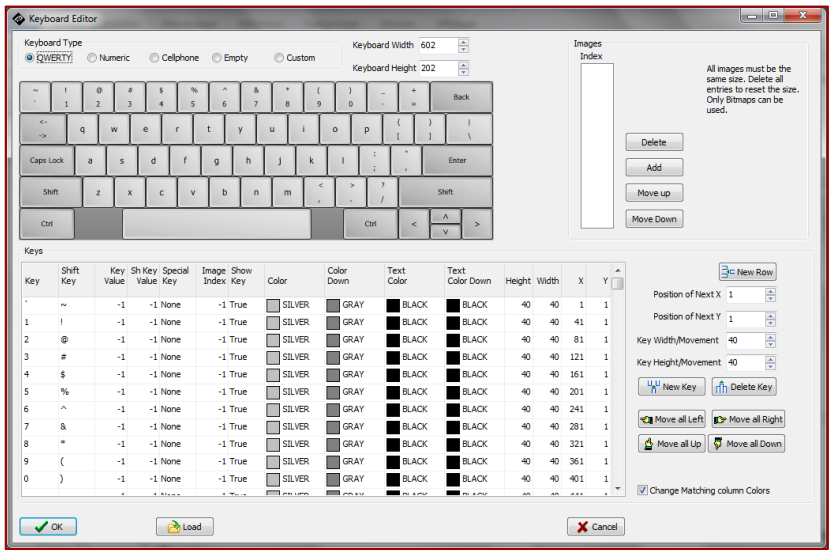
- Numeric keyboard

- And even a customised keyboard.

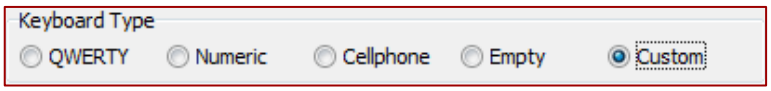
This object has one single event, onChanged, and sends the key pressed.
The different keyboards are selected by clicking on the **KeyboardType** property:



Click on the button  to launch the Keyboard Editor:

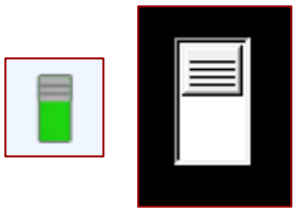


The Keyboard Editor allows you to select and customise the keyboard:



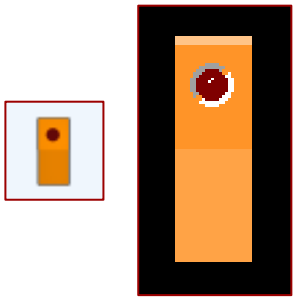
Note: For more information on the Keyboard object, please refer to the 4D Systems application note ViSi-Genie Customised Keyboard.

7.5.6. DIP Switch



The number of positions of the switch can be specified, 2 as shown or more.
This object has two events, onChanged and onChanging.

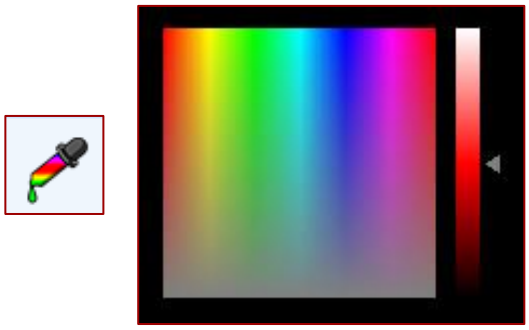
7.5.7. Rocker Switch



When on, the red LED is turned on. This object has two events, onChanged and onChanging.

Note: For more information on the Inputs objects, please refer to the 4D Systems application note ViSi-Genie Inputs.

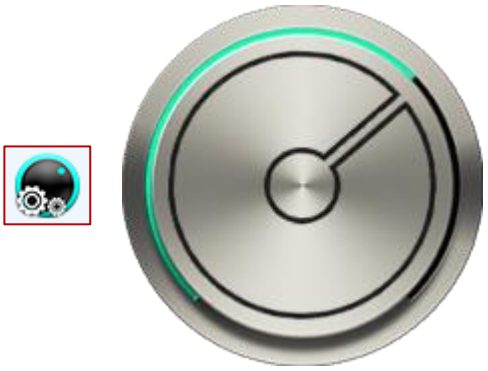
7.5.8. Color Picker



When touched at a certain point the color picker sends a corresponding message to the external host controller. The two-byte value contained by the message from a color picker represents the 16 bit color value in 565 format (5 bits of Red, 6 bits of Green, and 5 bits of Blue). On the other hand, if the host controller sends a message to the color picker, the colour contained by the message is indicated on the display. This object has one single event, onChanged, very useful to send the colour value.

Note: For more information on the color picker, please refer to the 4D Systems application note ViSi-Genie Color Picker.

7.5.9. Smart Knob



The smart knob is a highly customizable knob object that is accessible for **PRO** version users. This object allows user to create custom input knobs using multiple image frames. Each layer, except the face image, can be manipulated to move horizontally, vertically or rotate with respect to a specified point. It is also possible to use different static images and show them in order in a single layer. The smart knob works a lot similar to a regular knob.

For this object to properly work as designed, users would need to set the Min Angle and Max Angle properties of the object. The angle is measured from the negative y axis of the object (the center of the knob being the origin) counting positive angle by going clockwise.

Note: For more information on the Smart Knob object, please refer to the Smart Widgets Editor Manual.

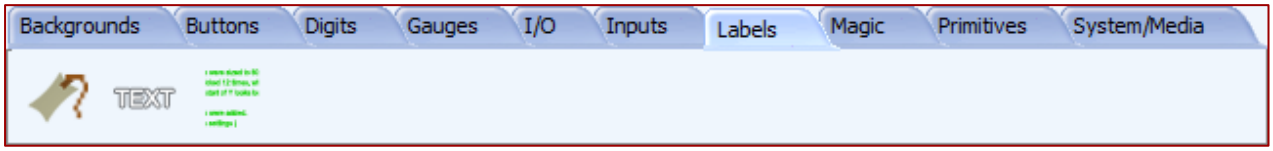
7.5.10. Smart Slider

The smart slider is a highly customizable slider object that is accessible for **PRO** version users. This object allows user to create custom output objects using multiple image frames. Each layer, except the face image, can be manipulated to move horizontally, vertically or rotate with respect to a specified point. It is also possible to use different static images and show them in order in a single layer. The smart knob works a lot similar to a regular slider.

For this object to properly work as designed, users would need to set the Min Offset and Max Max Offset properties of the object. Min Offset is the distance between the top (if vertical) or left (if horizontal) of the object and the nearest thumb end position. Max Offset is the distance between the bottom (if vertical) or right (if horizontal) of the object and the nearest thumb end position.

Note: For more information on the Smart Slider object, please refer to the Smart Widgets Editor Manual.

7.6. Labels Objects



The Labels pane offers three different objects to display text.

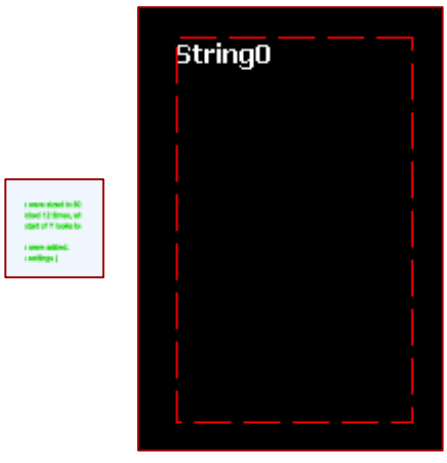
7.6.1. Label

This object has no event.

7.6.2. Static Text

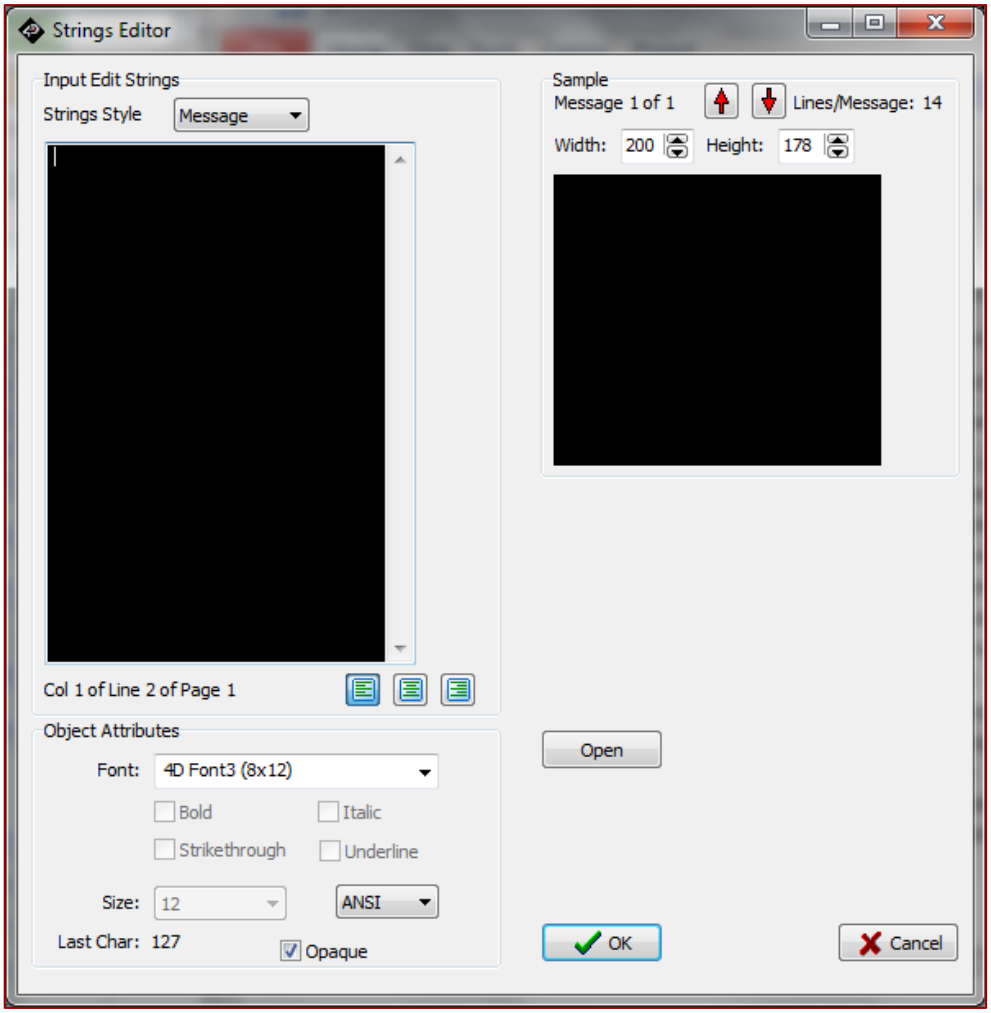
This object has no event.

7.6.3. Strings



This object displays a text.

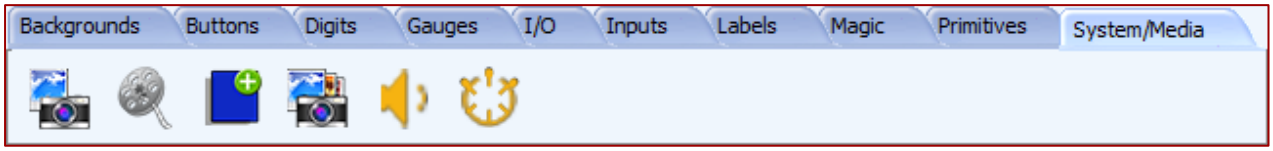
The text is defined by:



Font, size, ANSI or Unicode can be defined.
This object has no event.

Note: For more information on the Labels objects, please refer to the 4D Systems application note ViSi-Genie Labels, Text, and Strings.

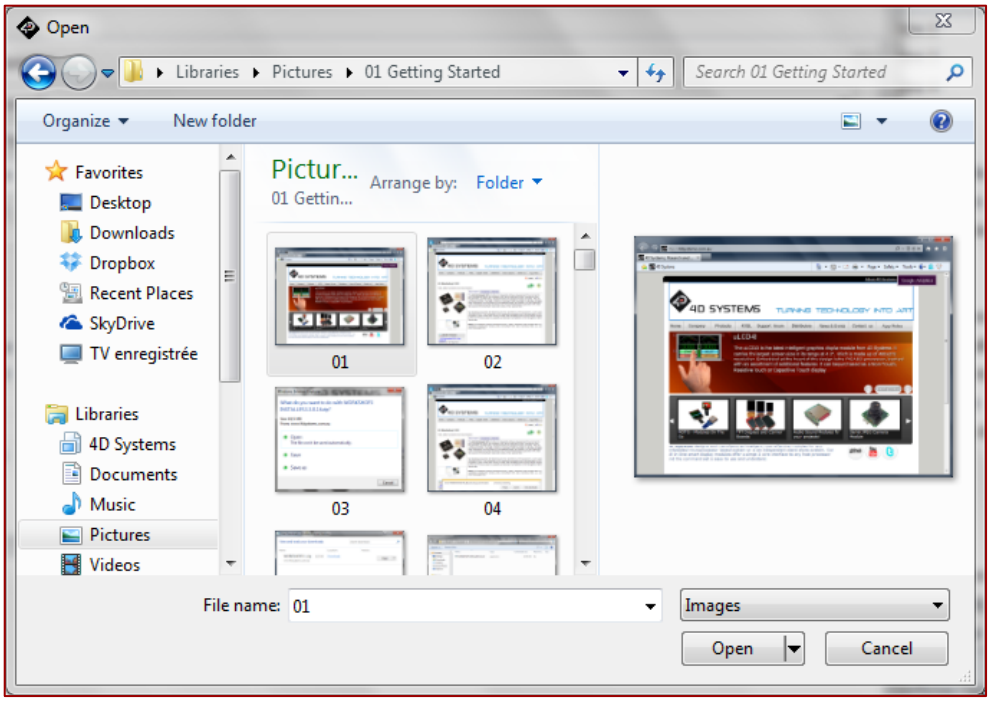
7.7. System/Media Objects



The System pane includes the form, image and video objects and two invisible objects, timer and sound.

7.7.1. Image

The image is selected through an Open window:

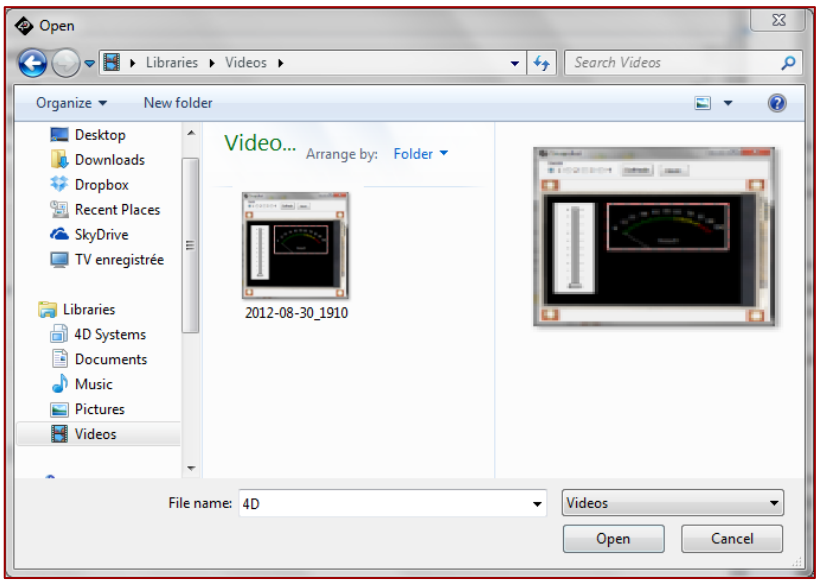


This object has no event.

Note: For more information on the Image object, please refer to the 4D Systems application note ViSi-Genie Show Image.

7.7.2. Video

The video is selected through an Open window:



This object has one single event, onChanged.

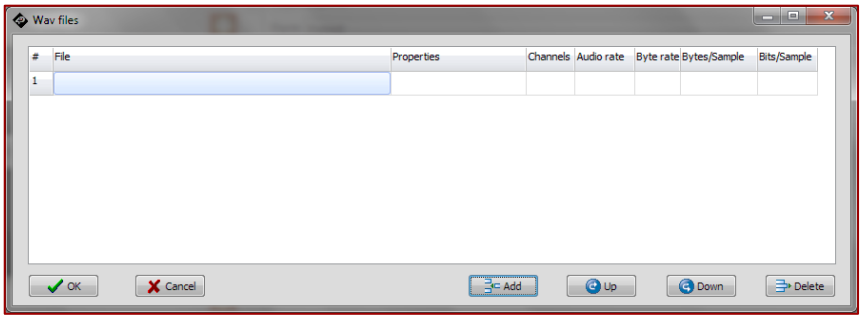
Note: For more information on the Video object, please refer to the 4D Systems application note ViSi-Genie Play Video.

7.7.3. Form

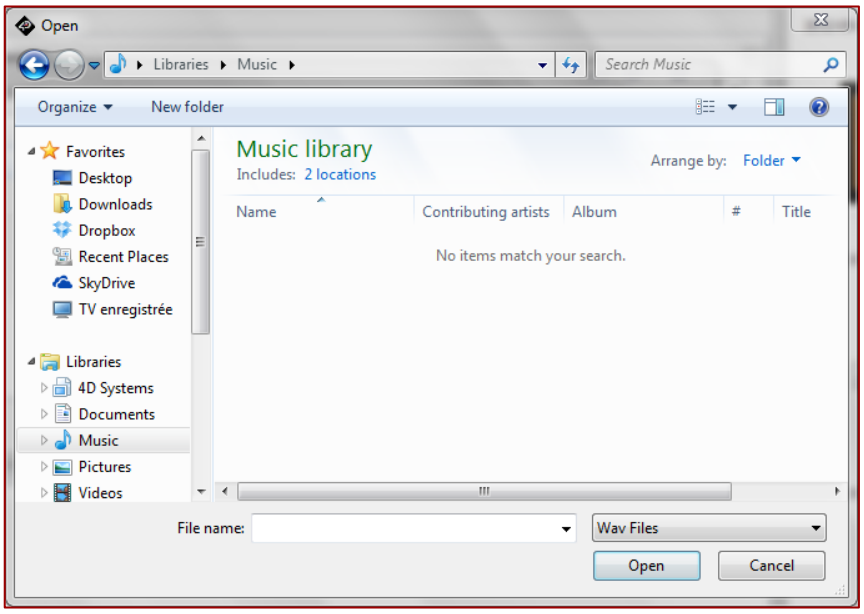
The Form creates a new empty form and adds it to the project.
This object has one single event, onActivate.

7.7.4. Sounds

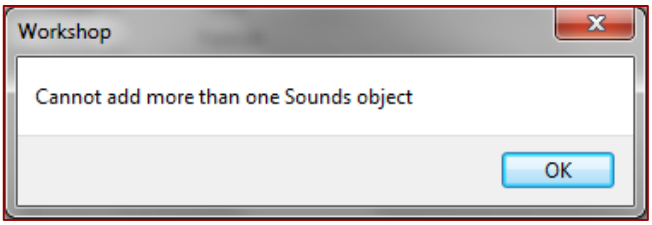
Sound is an invisible object. This object has two events, onPlayingChanged and onVolumeChanged.
The Sound object contains a list of sound files:



To add a sound file, click on **Add**: the sound file is selected through an Open window:



Files can be sorted by clicking on **Up** or **Down** and removed by clicking on **Delete**.
Only one Sound object can be added per project, but this sound object can contain multiple sound files.



Note: For more information on the Sound object, please refer to the 4D Systems application note ViSi-Genie Play Sound.

7.7.5. Timer

Timer is an invisible object. It raises an event, here every *1000 ms*.

This object has one single event, onTimer.

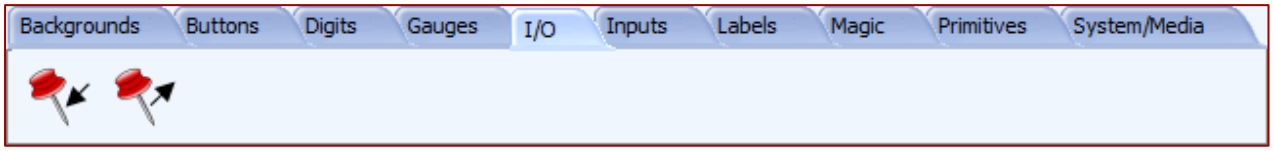
7.7.6. User Images

The user images object represents an easy way to build a slideshow by joining together a sequence of images in one place. The user provides the images and can use an input type object, such as a button or a slider, to make the user images object display the next or previous frame.

The user images object can also be made to behave as a video player with the use of a timer object. Each click of the timer will increment to the next frame. The user images object has one single event, onChanged, which is useful for sending the value of the current frame.

Note: For more information on the user images object, please refer to the 4D Systems application note ViSi-Genie User Images.

7.8. I/O



The I/O pane includes the pin input and the pin output.

Note: For more information on the I/O objects, please refer to the 4D Systems application note ViSi Genie Pin Input and Output for Picaso Display Modules or ViSi Genie Pin Input and Output for Diablo16 Display Modules.

7.8.1. Pin Input

The user can read the status of a specific pin when it's configured as a **PinInput** object. Multiple PinInputs can use the same pin. It is the users' responsibility to manage such usage in a reasonable way. This object has one single event, onChanged, which can cause either another output to be changed or a message to be sent to the host.

Do not set a pin to both input and output as undesirable results may occur. If you need to read an output pin from your host then use the normal read command for the PinOutput.

Note: The PinInput object (like the PinOutput) will always reside in Form0.

7.8.2. Pin Output

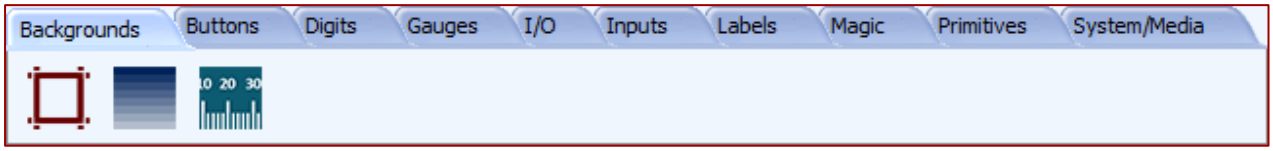
When a specific pin is configured as a **PinOutput** object, the user can control or pulse its output. Multiple PinOutputs can use the same pin. It is the users' responsibility to manage such usage in a reasonable way.

An input object such as a button can be linked to a PinOutput object. Logically, it makes sense to only connect a momentary button to a pulsed output pin and a toggled button to a non-pulsed output. Of course, it occasionally come in handy to be able to do the non-apparent, so you can set these options any way you like.

This object has one single event, onChanged, which can trigger another output to be changed or a message to be sent to the host. PinOutputs can be read by the host.

Note: The PinOutput object (like the PinInput) will always reside in Form0.

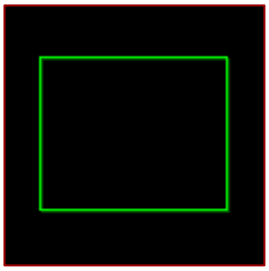
7.9. Backgrounds



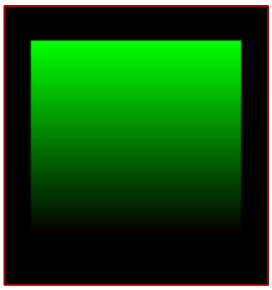
Background objects can be modified accordingly by the user.

Note: For more information on the backgrounds objects, please refer to the 4D Systems application note ViSi-Genie How to Add Background Objects.

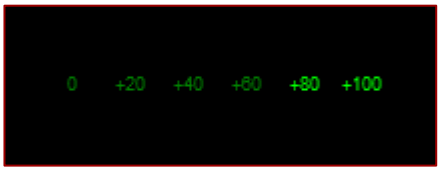
7.9.1. Border



7.9.2. Gradient

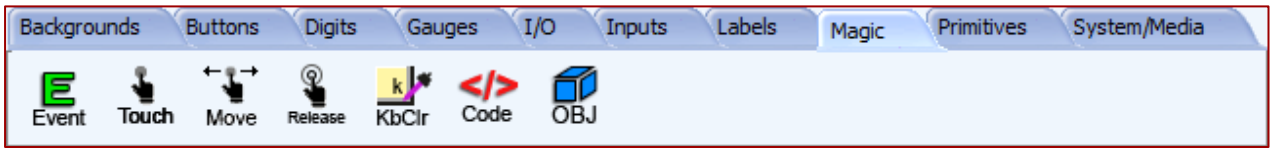


7.9.3. Scale



7.10. Magic Objects

Worskhop4 PRO has an extension called “Genie Magic”. Under Genie Magic there are additional objects that allow the user to add 4DGL code at various points in the program. These objects are called “Magic Objects”. These are found under the Magic pane.



Note: The Magic objects are available only in Workshop4 PRO. For more information on the Magic Objects, refer the ViSi-Genie Reference Manual and the 4D Systems application note ViSi-Genie: How to Add Magic Objects.

7.10.1. Magic Event

Events are usually triggered when input objects such as winbutton objects on the display are pressed. In the Genie environment of the standard Workshop IDE, the OnChanged event property of a winbutton, for example, can be configured to toggle the state of a LED object. In Workshop Pro, it is now possible for the user to create a custom event, through 4DGL coding, in the form of a Magic Event object. For instance, a winbutton, when pressed, can make a LED object blink ten times.

7.10.2. Magic Touch

A Magic Touch object contains a 4DGL routine that is executed the moment that a “TOUCH_PRESSED” action is detected on the display. For example, the description for a button with an icon can be printed the moment that it is pressed.

7.10.3. Magic Move

A Magic Move object contains a 4DGL routine that is executed the moment that a “TOUCH_MOVING” action is detected on the display. For example, a line can be drawn along the path of a stylus as it is being dragged across a large button.

7.10.4. Magic Release

A Magic Release object contains a 4DGL routine that is executed the moment that a “TOUCH_RELEASED” action is detected on the display. For example, the printed description for a button with an icon can be “erased” the moment that the button is pressed.

7.10.5. Magic Keyboard and Color Picker Event

Similar to the Magic Event object, the Magic Keyboard + Color Picker Event object is a custom event. More specifically, the Magic Keyboard + Color Picker Event is for handling events from keyboard and color picker objects.

7.10.6. Magic Code

A Magic Code object allows the user to insert custom 4DGL code into specific locations inside the Genie project. For example, a counter variable can be declared and initialized in a Magic Code object inserted to the location “Constant/Global/Data”. This variable can then be accessed and used by another Magic Code object inserted at the location “MainLoop”.

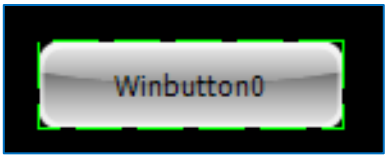
7.10.7. Magic Object

A Magic Object allows the user to write a custom handler for messages received from the serial port.

7.11. Selection Tool

The arrow is used to deselect an object.

To select an object, just click on it: green or red dotted lines appear.



To deselect an object, just click again: the dotted lines disappear.

8. ViSi-Genie Communications Protocols

The ViSi-Genie display platform offers a serial communications protocol called the **Genie Standard Protocol**. The protocol provides access to a majority of the display's features and gives the host detailed information on the current state of all the objects used in the display application.

The **Genie Standard Protocol** provides a simple yet effective interface between the display and the host controller and all communications are reported over this bidirectional link. The protocol utilises only a handful of commands and is simple and easy to implement.

Serial data settings are:

8 Bits, No Parity, 1 Stop Bit.

The baud rate for the display is selected from the Workshop Genie project. The user should match the same baud rate on the host side.

Note: RS-232 handshaking signals (i.e., RTS, CTS, DTR, and DSR) are not supported by the ViSi-Genie protocols. Instead, only the RxD (received data), TxD (transmitted data), and signal ground are used.

Objects are drawn on the display in the order they are created in the Workshop project. If Image objects are to be used for the background and other objects on top, then the image objects must be created and added first. Also note this only applies to non-active Image objects, other active objects should not be added on top of each other.

8.1. Genie Standard Protocol

This section describes the Genie Standard Protocol in detail.

8.1.1. Protocol Definitions

The commands and parameters are sent and received using a very simple messaging structure. The message consists of a command byte, command parameters, and a checksum byte. The checksum ensures some the integrity of the message. The following figure shows the organisation of the message.

CMD	PARAM (1 to N bytes)	CHKSUM

- **CMD:** This byte indicates the command code. Some commands will have more parameters than others. The table below outlines the available commands and their relevant parameters.
- **PARAM:** Parameter bytes (variable); a variable number of parameter bytes (between 1 to N) that contains information pertaining to the command. Refer to the command table below.
- **CHKSUM:** Checksum byte; this byte is calculated by taking each byte and XOR'ing all bytes in the message from (and including) the CMD byte to the last parameter byte. Then, the result is appended to the end to yield the checksum byte.

Note: If is correct, check byte plus the sum of all the other bytes in the message will give a result of 0.

8.1.2. Command and Parameters Table

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
WRITE_MAGIC_BYTES	0X08	Object Index	Length	Array (1 byte values)			Checksum
WRITE_MAGIC_DBYTES	0x09	Object Index	Length	Array (2 byte values)			Checksum
REPORT_MAGIC_EVENT_BYTES	0x0A	Object Index	Length	Array (1 byte values)			Checksum
REPORT_MAGIC_EVENT_DBYTES	0x0B	Object Index	Length	Array (2 byte values)			Checksum

* Magic commands are only available when using Genie PRO.

8.1.3. Command Set Messages

The ViSi-Genie Reference Manual provides detailed information intended for programmers of the Host Controller. It contains the message formats of the commands that comprise the ViSi-Genie protocol. New commands may be added in future to expand the protocol.

8.1.4. Acknowledgement Bytes Table

ACK	Acknowledge byte (06hex); this byte is issued by the Display to the Host when the Display has correctly received the last message frame from the Host. The transmission message for this is a single byte: 06hex
NAK	Not Acknowledge byte (15hex); this byte is issued by the receiver (Display or Host) to the sender (Host or Display) when the receiver has not correctly received the last message frame from the sender. The transmission message for this is a single byte: 15hex

8.1.5. Genie Advanced Protocol

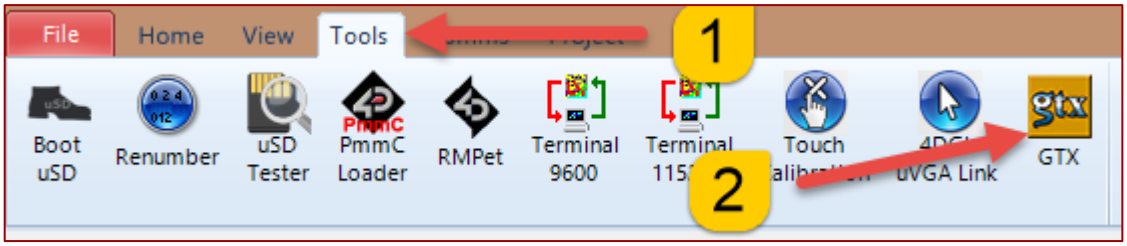
Genie advanced protocol allows managing multiple screens will be released soon.

8.2. Object Types Table

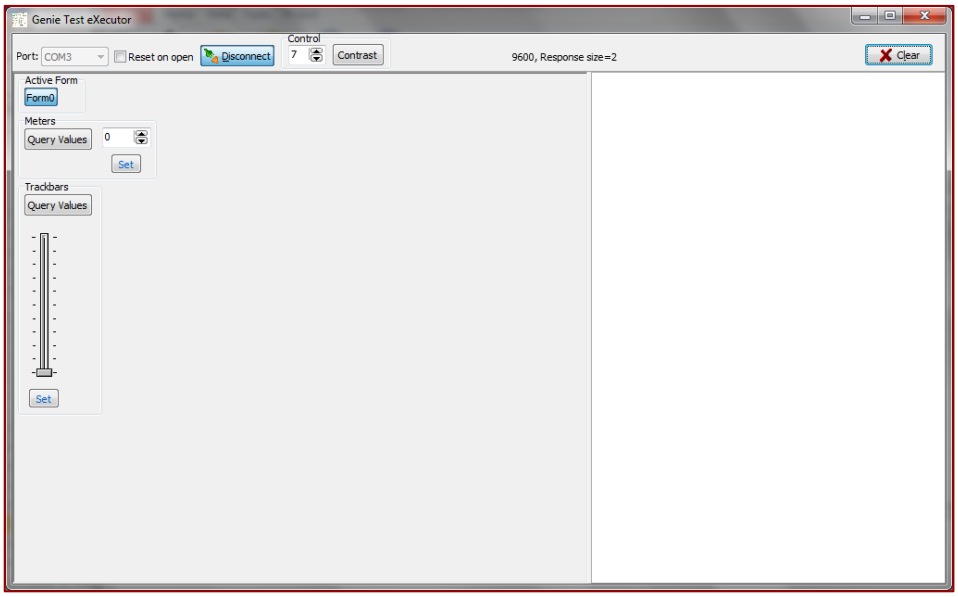
Object	ID	Input	Output	Notes
Dipswitch	0 (0x00)	✓	✓	
Knob	1 (0x01)	✓	✓	
Rockerswitch	2 (0x02)	✓	✓	
Rotaryswitch	3 (0x03)	✓	✓	
Slider	4 (0x04)	✓	✓	
Trackbar	5 (0x05)	✓	✓	
Winbutton	6 (0x06)	✓	✓	
Angularmeter	7 (0x07)		✓	
Coolgauge	8 (0x08)		✓	
Customdigits	9 (0x09)		✓	
Form	10 (0x0A)		✓	Used to set the 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)		✓	
Spectrum	24 (0x18)		✓	
Scope	25 (0x19)		✓	
Tank	26 (0x1A)		✓	
UserImages	27 (0x1B)		✓	
PinOutput	28 (0x1C)		✓	
PinInput	29 (0x1D)	✓		
4Dbutton	30 (0x1E)	✓	✓	
AniButton	31 (0x1F)	✓	✓	
ColorPicker	32 (0x20)	✓	✓	
UserButton	33 (0x21)	✓	✓	
MagicObject	34 (0x22)		✓	Only available when using Genie PRO
SmartGauge	35 (0x23)		✓	Only available when using Genie PRO
SmartSlider	36 (0x24)	✓	✓	Only available when using Genie PRO
SmartKnob	37 (0x25)	✓	✓	Only available when using Genie PRO

9. Integrated Debugger

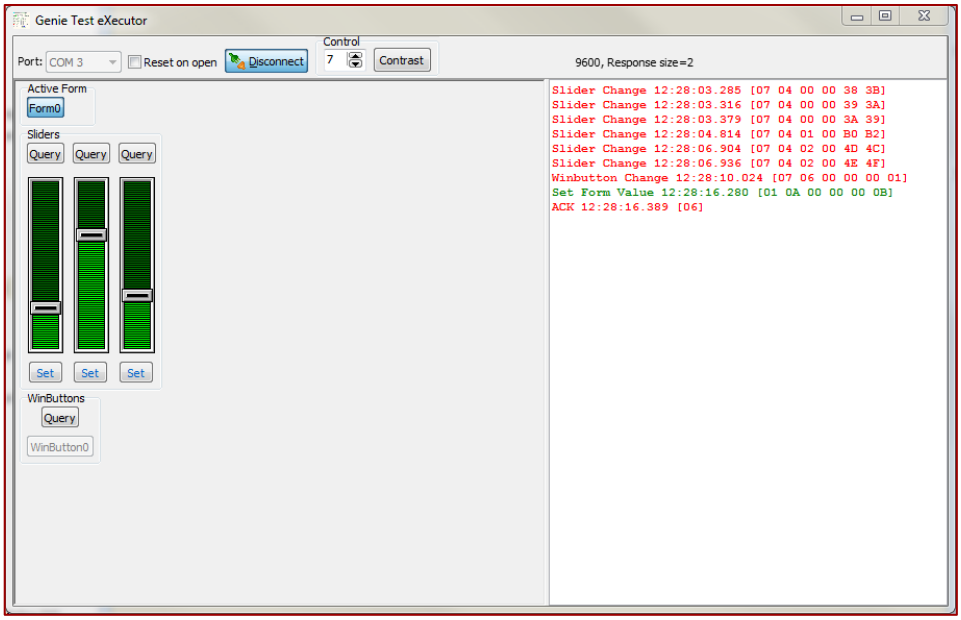
The integrated debugger of Workshop4 is called **Genie Test Executor** or GTX.
To launch the debugger, click on the **GTX** button available on the menu **Tools**.



A new screen appears, with the form and objects we have defined previously:



Just try to move the track-bar and press Set: the value is sent to the screen. Pressing **Query Values** reads the value from the screen's track-bar.



The white area on the right displays:

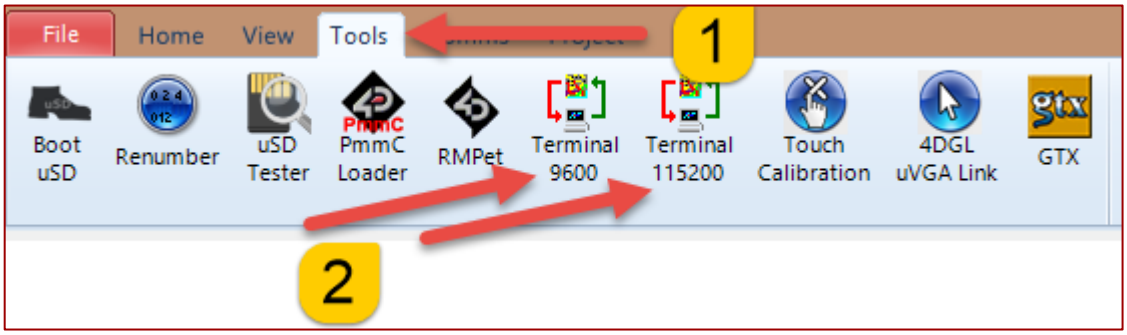
- In **green** the messages sent to the screen;
- And in **red** the messages received from the screen:

```
Slider Change 12:28:03.285 [07 04 00 00 38 3B]
Slider Change 12:28:03.316 [07 04 00 00 39 3A]
Slider Change 12:28:03.379 [07 04 00 00 3A 39]
Slider Change 12:28:04.814 [07 04 01 00 B0 B2]
Slider Change 12:28:06.904 [07 04 02 00 4D 4C]
Slider Change 12:28:06.936 [07 04 02 00 4E 4F]
Winbutton Change 12:28:10.024 [07 06 00 00 00 01]
Set Form Value 12:28:16.280 [01 0A 00 00 00 0B]
ACK 12:28:16.389 [06]
```

All values are in hexadecimal.

10. Communication Terminal

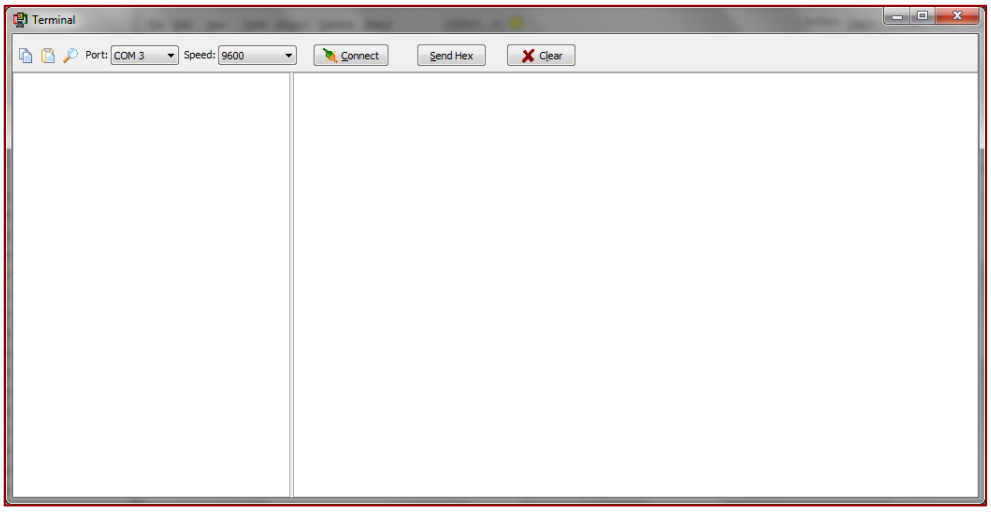
An alternative to the debugger is the terminal. To launch the Terminal, select the **Tools** menu…



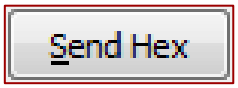
…and

- Click '**Terminal connect 9600**' to open the currently selected com port at 9600 baud in the Terminal program.
- Click '**Terminal connect 115200**' to open the currently selected com port at 115200 baud in the Terminal program.

A new screen appears:



To send the commands on hexadecimal format, press

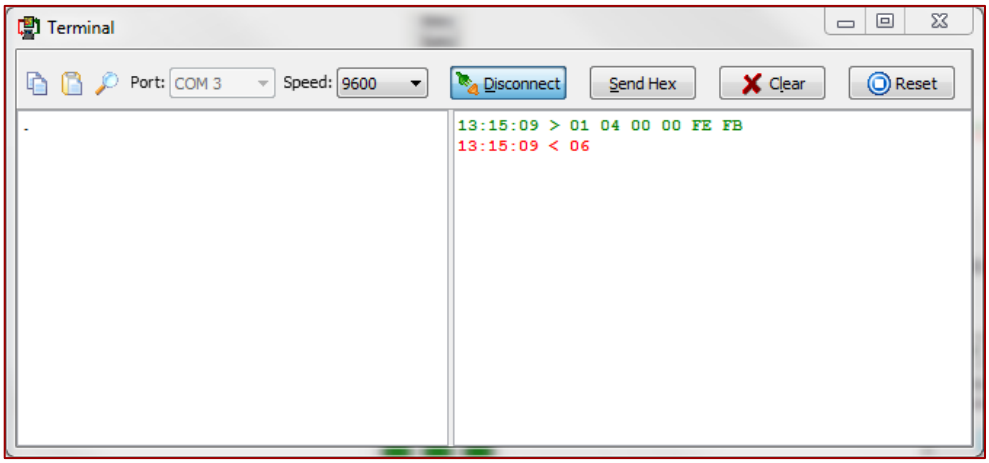


The commands sent by the host and the messages sent by the screen are the same as with the **Genie Test Executor** debugger.

The white area on the right displays

- In **green** the messages sent to the screen;
- And in **red** the messages received from the screen:
-

Here, the command *Set Slider0 to value 0x17* is sent, or **04 00 17** displayed in green on the terminal window.



And the screen answers with the **0x06** successful acknowledgement, displayed on red on the terminal window.

11. Application Notes

For a more detailed presentation of the objects with examples, please refer to the application notes that can be found at http://www.4dsystems.com.au/appnotes

Note: For an exhaustive reference on ViSi-Genie objects, please refer to the ViSi-Genie Reference Manual.

12. Revision History

Revision	Revision Content	Revision Date
1.0	First Release	21/03/2014
1.1	Fixed FONT references which were incorrectly copied from Picaso	04/05/2014
1.2	Updated image in Section 2.2	07/05/2014
1.3	Fixed typo in putstr function reference (was putStr)	01/10/2014
1.4	Fixed a few typos regarding Contrast. All Diablo16 modules are 0-15	30/10/2014
1.5	Added information for file_LoadImageControl. Updated control block size in file_Mount. Added information relating to Set Font and uSD based fonts. Added note about restriction of clipping command. Added information about the use of TRANSPARENCY.	22/12/2014
1.6	Added max write size to “File Write” command. Fixed FontIDs for deja fonts	29/06/2015
2.0	Updated formatting and contents	01/05/2017
2.1	Updated formatting	05/04/2019
2.2	Fixed broken links to use Resource Centre counterparts	27/02/2024

13. 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 makes 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.

14. Contact Information

For Technical Support: www.4dlabs.com.au/support
For Sales Support: sales@4dlabs.com.au
Website: www.4dlabs.com.au