



4D SYSTEMS
TURNING TECHNOLOGY INTO ART

Workshop 4 - ViSi
User Guide

USER GUIDE

Document Date: 22nd December 2013
Document Revision: 1.3

Contents

1. Description	4
2. Application Overview	4
3. Setup Procedure	4
4. Launch Workshop 4	5
5. Create a New Project	6
6. Select ViSi	9
7. The Main Screen	10
7.1. Area 1: Menus.....	11
7.2. Area 2: Ribbon with Icons	11
7.3. Area 3: Code.....	12
7.4. Area 4: Form and WYSIWYG Screen	12
7.5. Area 5: Object Inspector	13
7.6. Area 6: Message Window	13
7.7. Move the Form and Object Inspector.....	13
8. A First ViSi Project	15
8.1. Adding Graphical Objects.....	15
8.2. Customizing Graphical Objects	16
8.3. Changing Between Object Properties	17
8.4. Inserting a Layout into the User Application Program.....	18
8.5. Finalising a Program for Download	19
8.6. Downloading an Application	19
8.7. Example Application Program.....	19
9. Objects	21
9.1. Buttons Object	21
9.2. Digits Objects	22
9.3. Gauges Objects	23
9.4. Primitives Objects	25
9.5. Inputs Objects.....	27
9.6. Labels Objects.....	31
9.7. System/Media Objects.....	33
9.8. Selection Tool.....	36
10. Connect the Screen	37
11. Insert the Micro-SD Card	38
12. Communication Terminal	39

13. Application Notes42

14. Revision History43

15. Legal Notice44

16. Contact Information.....44

1. Description

This Application Note is dedicated to explaining the new 4D-ViSi Software Tool, which is part of the Workshop 4 IDE suite.

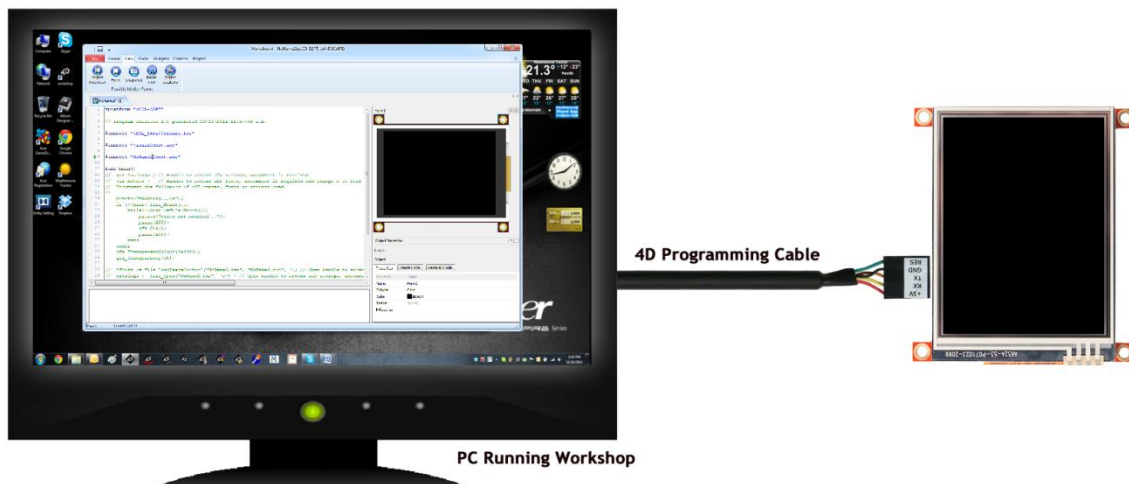
An overview of its basic functionality and uses will be explored. In order to use 4D-ViSi, the following items are required:

- Any 4D Systems Display Module
- 4D Programming Cable or Adaptor
- A micro-SD memory card
- 4D Systems Workshop 4 IDE

2. Application Overview

It is often difficult to design a graphical display without being able to see the immediate results of the application code. 4D-ViSi is the perfect software tool that allows the user to see the instant results of their desired graphical layout.

Additionally, there is a selection of inbuilt dials, gauges and meters that can simply be dragged and dropped onto the simulated module display. From here, each can have properties edited and at the click of a button, all relevant code is produced in the user program. Each feature of 4D-ViSi will be outlined with examples below.

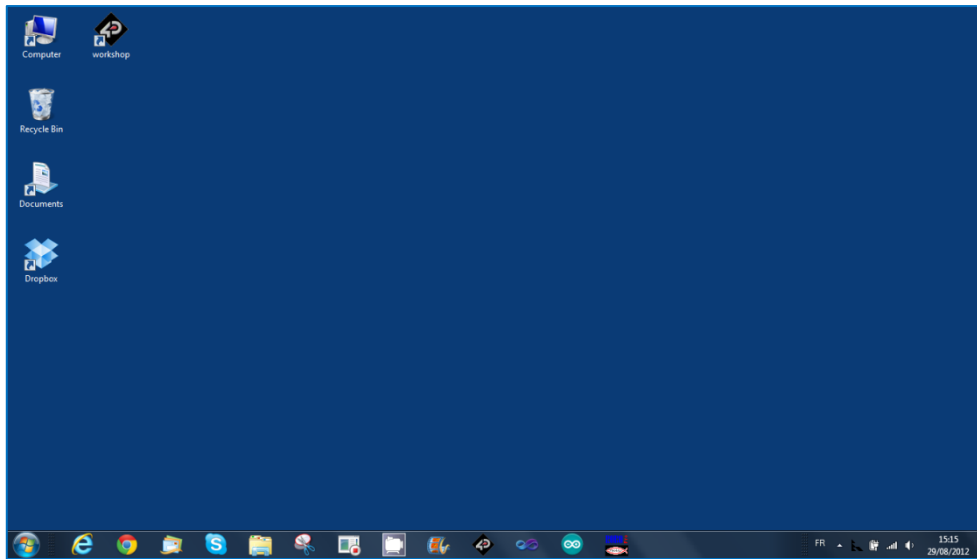


3. Setup Procedure

Firstly, you will need to download 4D-ViSi. It can be found from within the 4D Workshop 4 IDE product page on the 4D Systems website, www.4dsystems.com.au

4. Launch Workshop 4

There is an alias for 4D Workshop on the desktop:



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



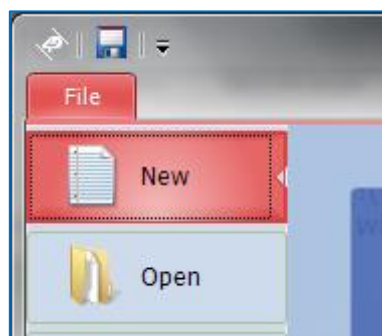
5. Create a New Project

Workshop 4 opens and displays the **Recent** page:



To create a new program, there are multiple options:

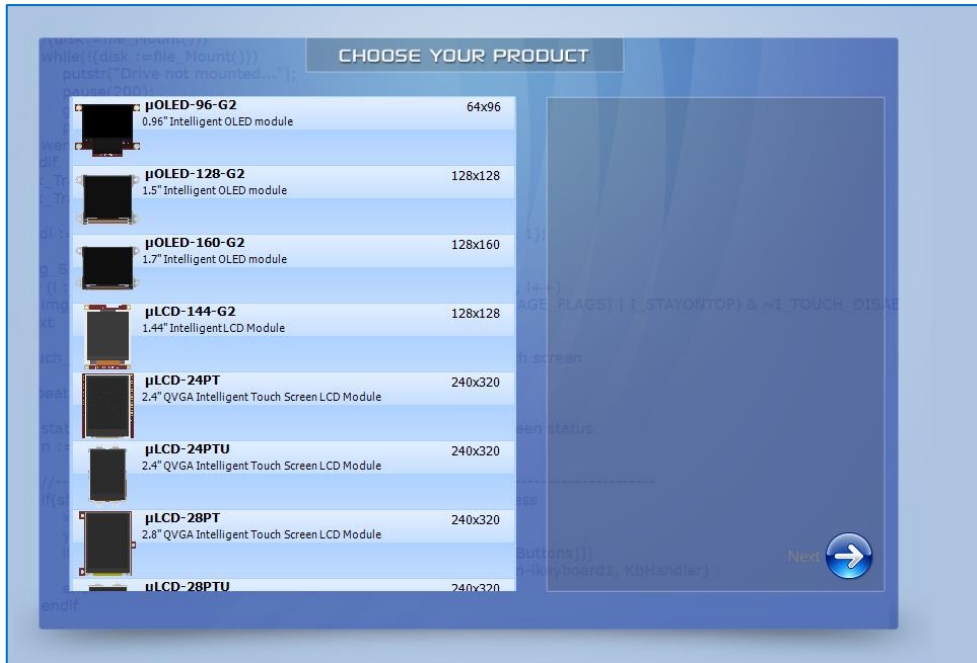
- Click on the top left-most icon **New**



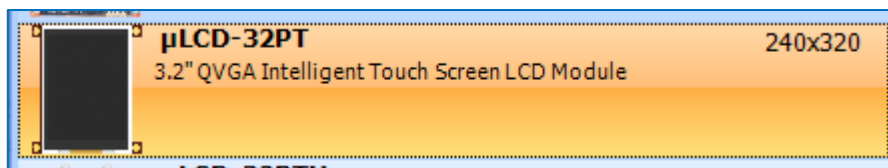
- Click on the icon close to **Create a New Project** on top or, if the settings have been already defined, click on the icon close to **Create a New Project** on bottom:



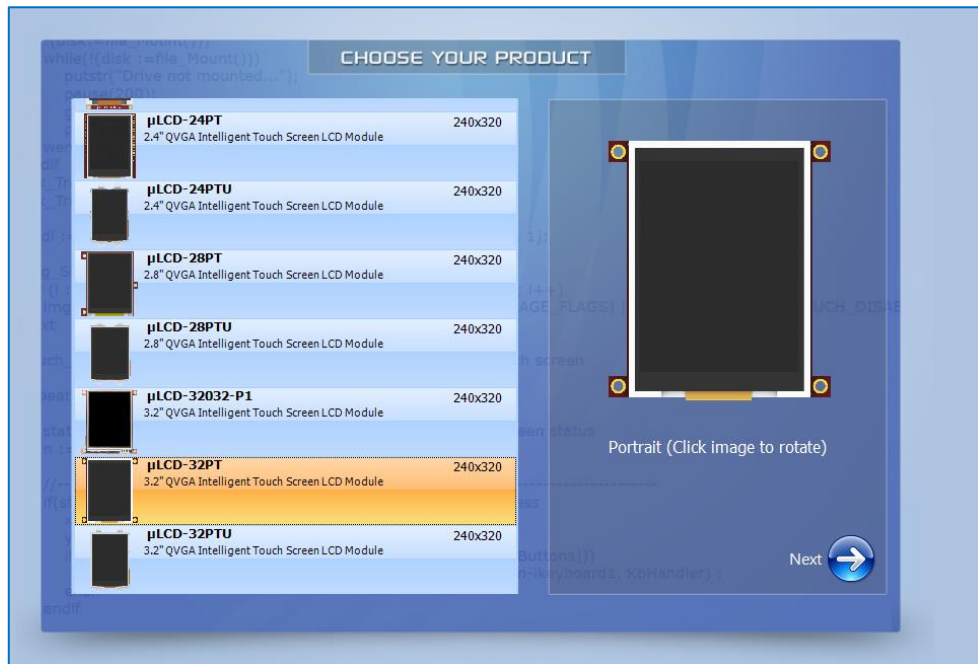
All those options update the main window with the selection of the screen:



Select the screen, here the μLCD-32PT:

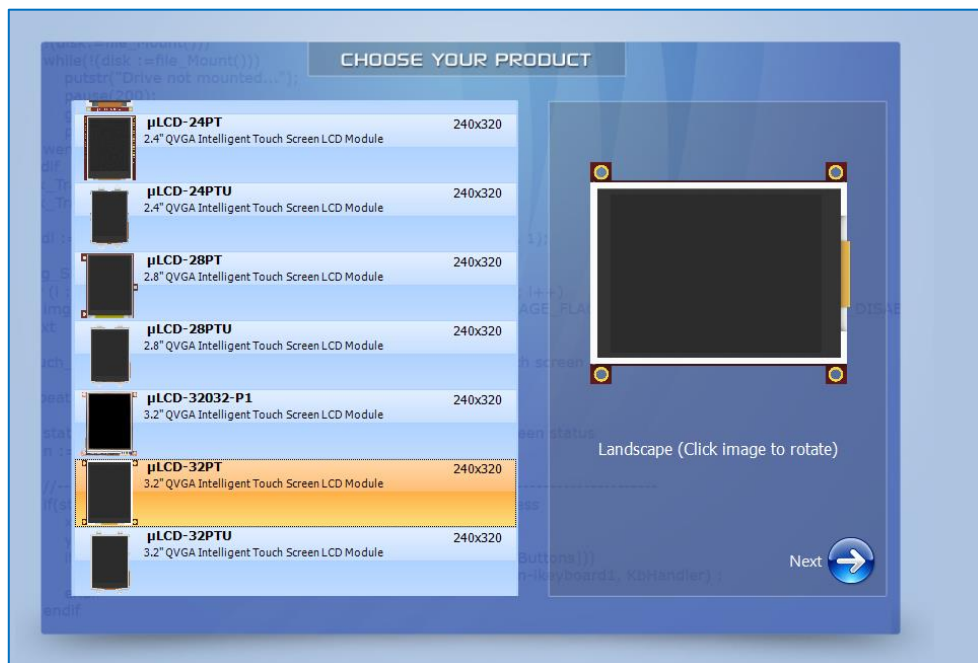


The selected screen is displayed:



Orientation is portrait by default.

To set it to landscape, just click on the image of the screen to rotate it:



Press **Next** to proceed:



6. Select ViSi

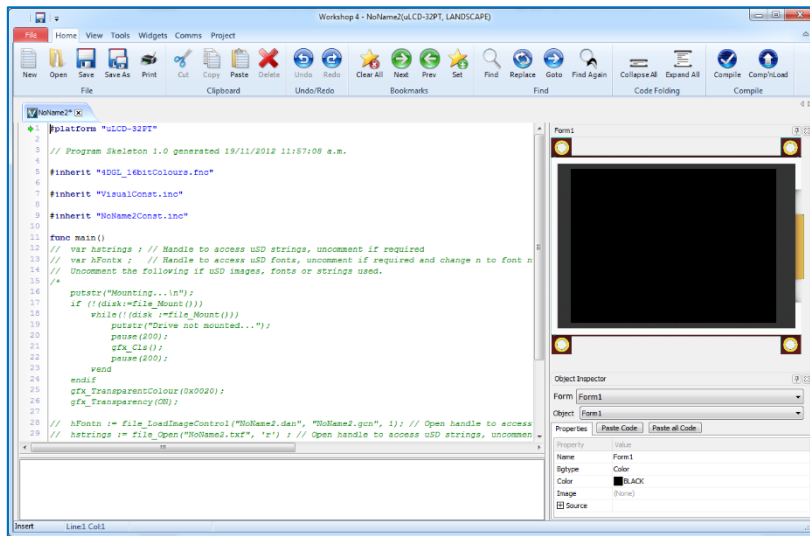
The main window now asks for the kind of project:



To select ViSi Genie, just click on the blue arrow:

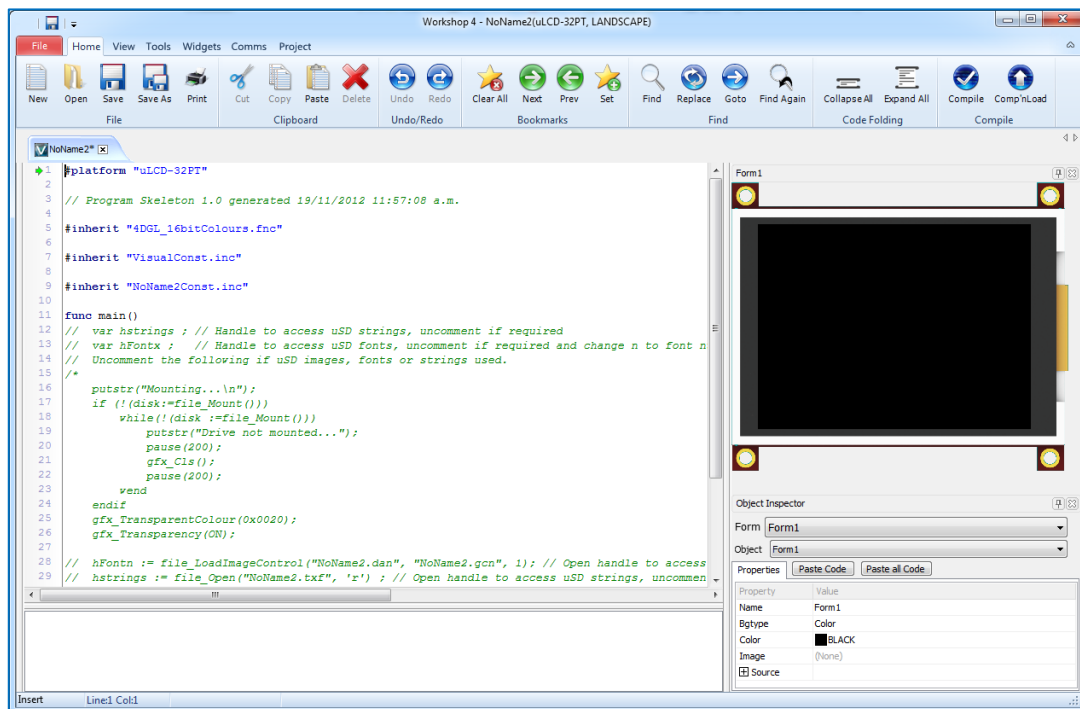


The development environment is now displayed:



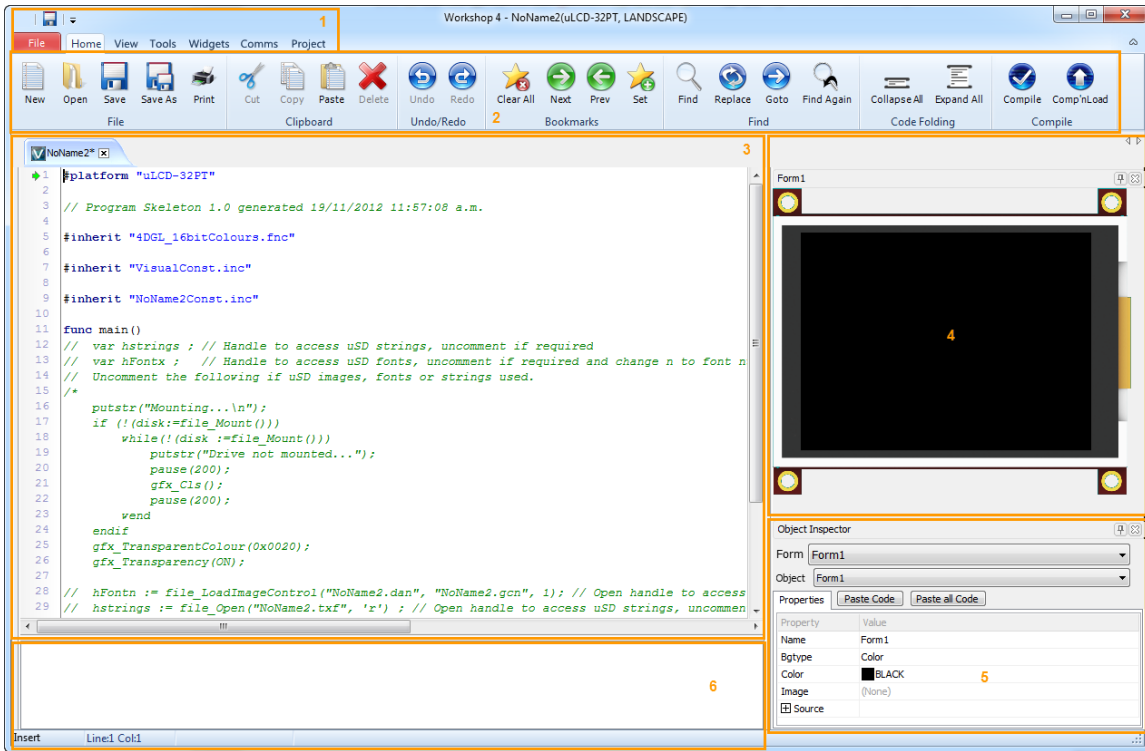
7. The Main Screen

The main screen appears:



Let's detail the different areas.

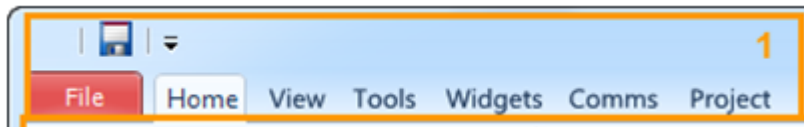
There are six different areas, from left to right, for top to bottom:



1. Menus;
2. Ribbon with icons;
3. Code;
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.

7.1. 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.

7.2. 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 **Compile** button compiles the project while the **Comp'nLoad** button compiles and uploads it to the screen.

7.3. Area 3: Code

On top of the code, the open projects are displayed:



```

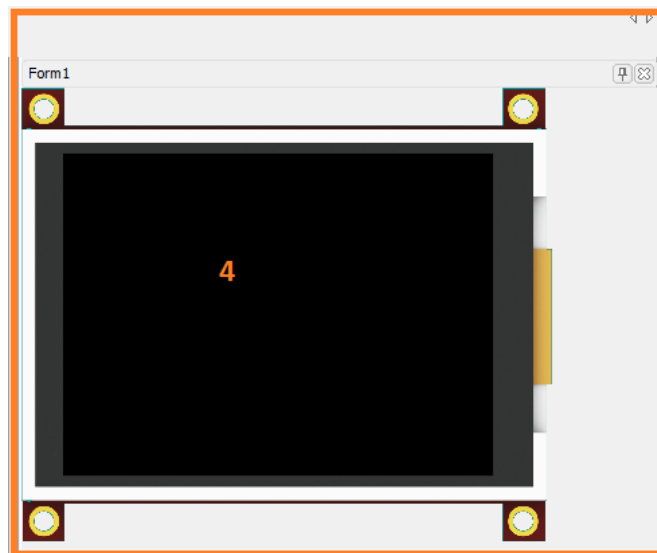
1 #platform "uLCD-32PT"
2
3 // Program Skeleton 1.0 generated 19/11/2012 11:57:08 a.m.
4
5 #inherit "4DGL_16bitColours.fnc"
6
7 #inherit "VisualConst.inc"
8
9 #inherit "NoName2Const.inc"
10
11 func main()
12 // var hstrings ; // Handle to access USD strings, uncomment if required
13 // var hFontx ; // Handle to access USD fonts, uncomment if required and change n to font n
14 // Uncomment the following if USD images, fonts or strings used.
15 /*
16     putstr("Mounting...\n");
17     if (!(disk:=file_Mount()))
18         while(!(disk :=file_Mount()))
19             putstr("Drive not mounted...");
20             pause(200);
21             gfx_Cls();
22             pause(200);
23     vend
24 endif
25 gfx_TransparentColour(0x0020);
26 gfx_Transparency(ON);
27
28 // hFontn := file_LoadImageControl("NoName2.dan", "NoName2.gcn", 1); // Open handle to access
29 // hstrings := file_Open("NoName2.txf", 'r') ; // Open handle to access USD strings, uncommen

```

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

7.4. Area 4: Form and WYSIWYG Screen

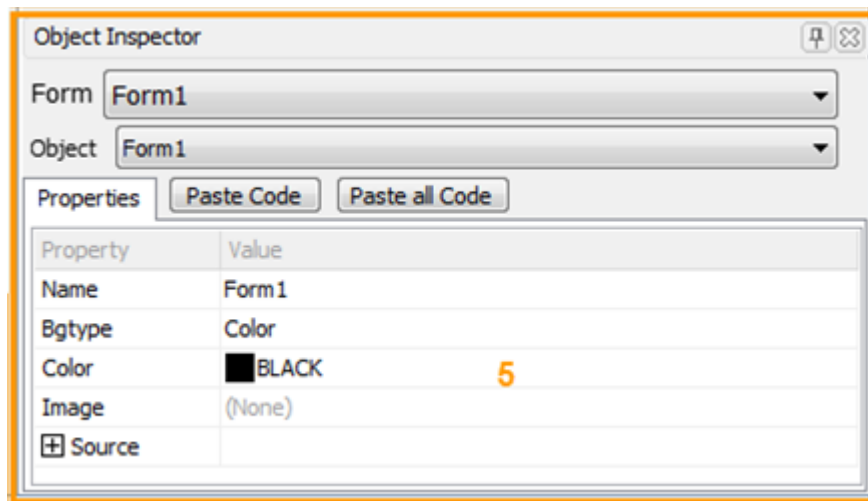
The form represents a What-You-See-Is-What-You-Get (WYSIWYG) screen.



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.

7.5. Area 5: Object Inspector

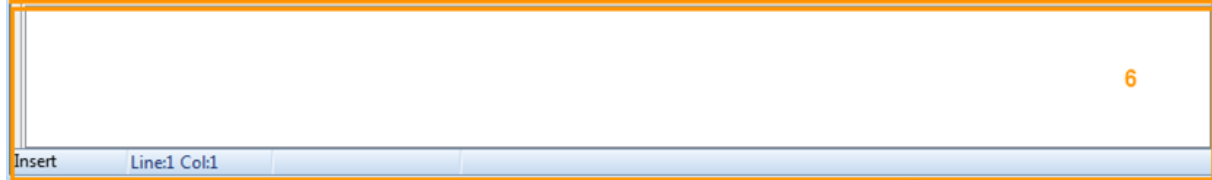


The object inspector provides all the information on the selected object:

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

7.6. Area 6: Message Window

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

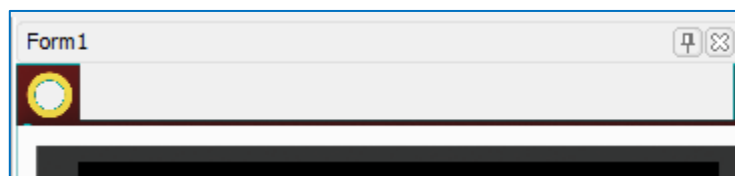


Before starting using the Workshop 4, we need to connect the screen and prepare a micro-SD card.

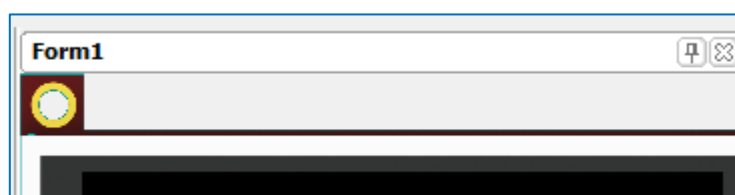
7.7. Move the Form and Object Inspector

The Form and Object Inspector can be moved for a customised interface.

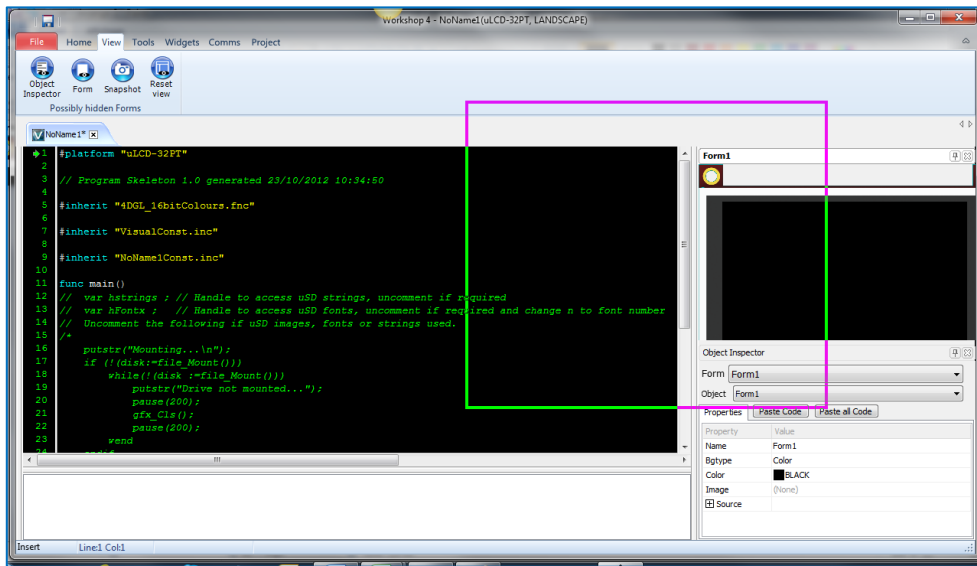
Click on the title bar of the form...



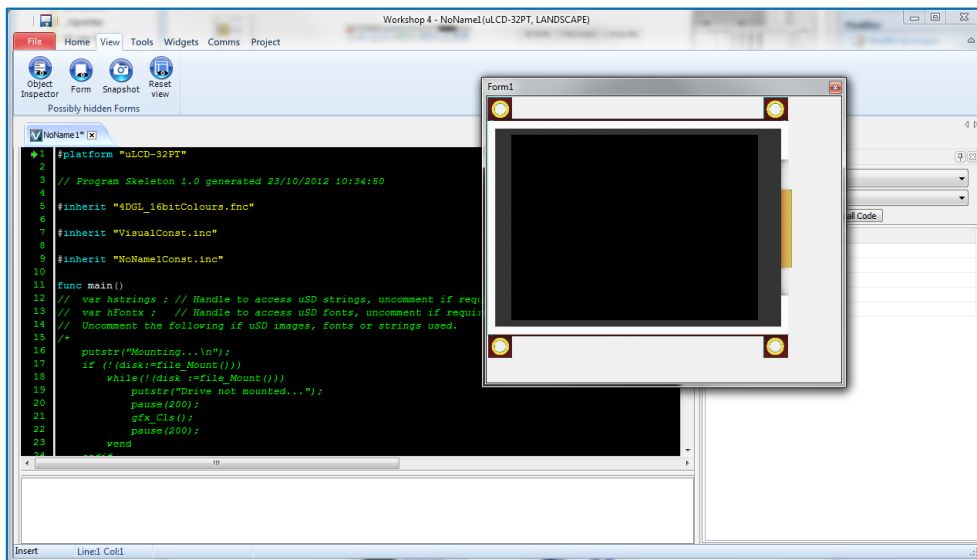
...to activate it...



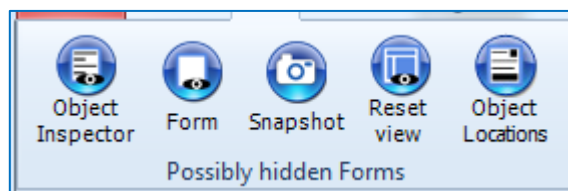
...then click-and-drag to place the form on the desire place...



...and finally release the button.



In case the Form and Object Inspector are hidden, just click on the relevant button on the **View** menu to display them again:



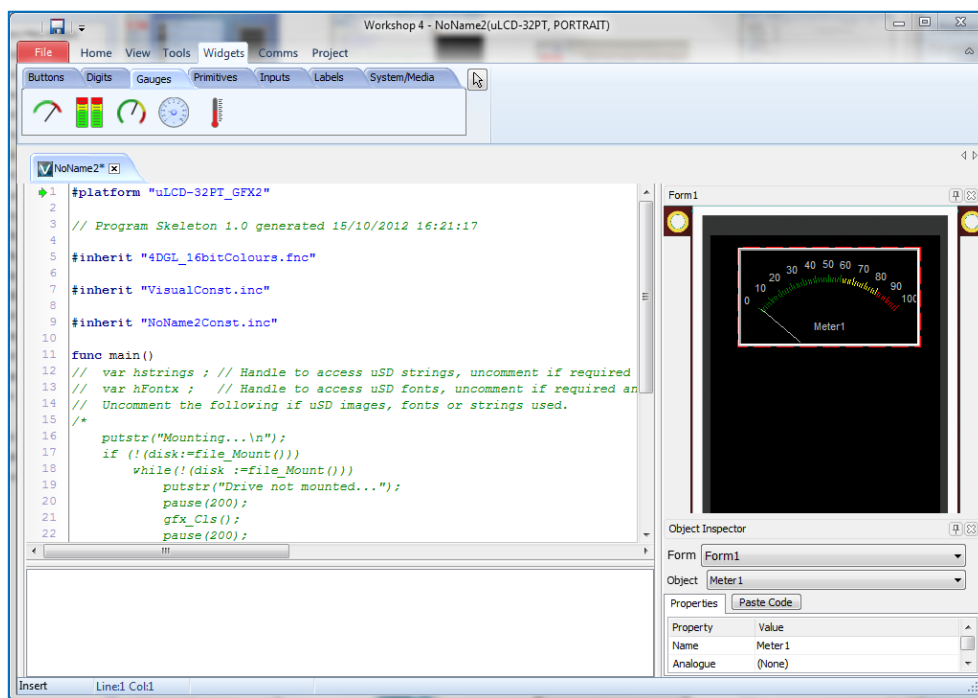
8. A First ViSi Project

8.1. Adding Graphical Objects

Become familiar with the 4D-ViSi layout by cycling through the tabs in the **Widgets** menu. Each tab contains a different category of graphical objects.

Try placing a meter gauge by selecting the **Gauges** tab and then click on the first dial icon, followed by clicking anywhere on the display area. The meter will be placed in the area, which is now ready for positioning and customization if necessary.

The following snapshot shows the gauge positioned at the top of the display area with editable properties below.



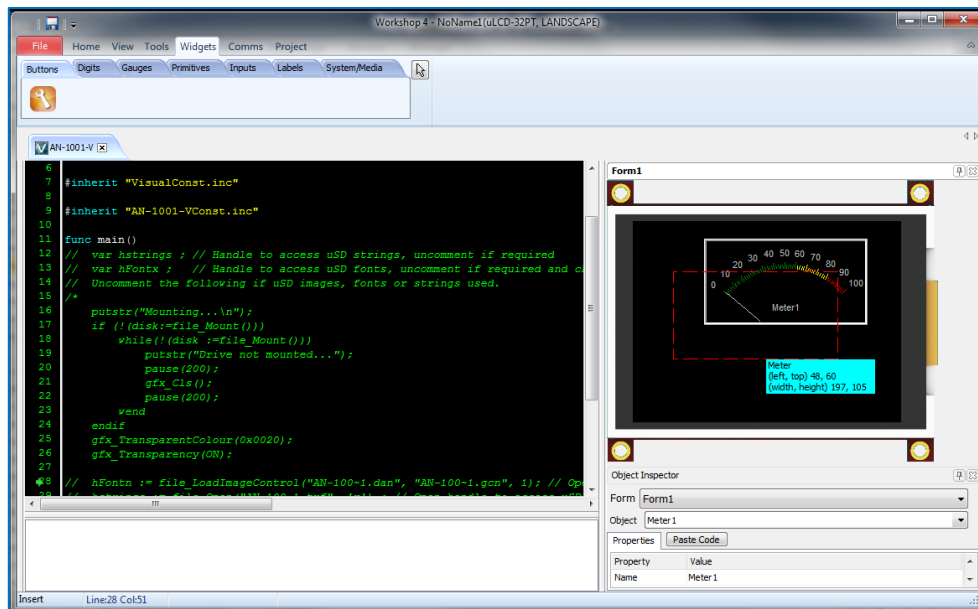
8.2. Customizing Graphical Objects

Now that the object has been placed, it can be fully customized to the desired layout by the user. Each object can be edited in one of two ways; either interactively on the display, or manually using the various fields underneath the display.

Take some time to experiment with the properties available to change the way an object appears on the screen. The easiest approach to achieving the optimal layout is by adjusting the size and position of the object directly on the display by moving the cursor over each entity. It should be noted that the background display or overall binding entity is referred to as **Form1**.

Form1 can be edited by selecting it from the dropdown box in the **Object Inspector**. Once happy, move onto changing features such as colours, fonts, captions,... in the property fields below.

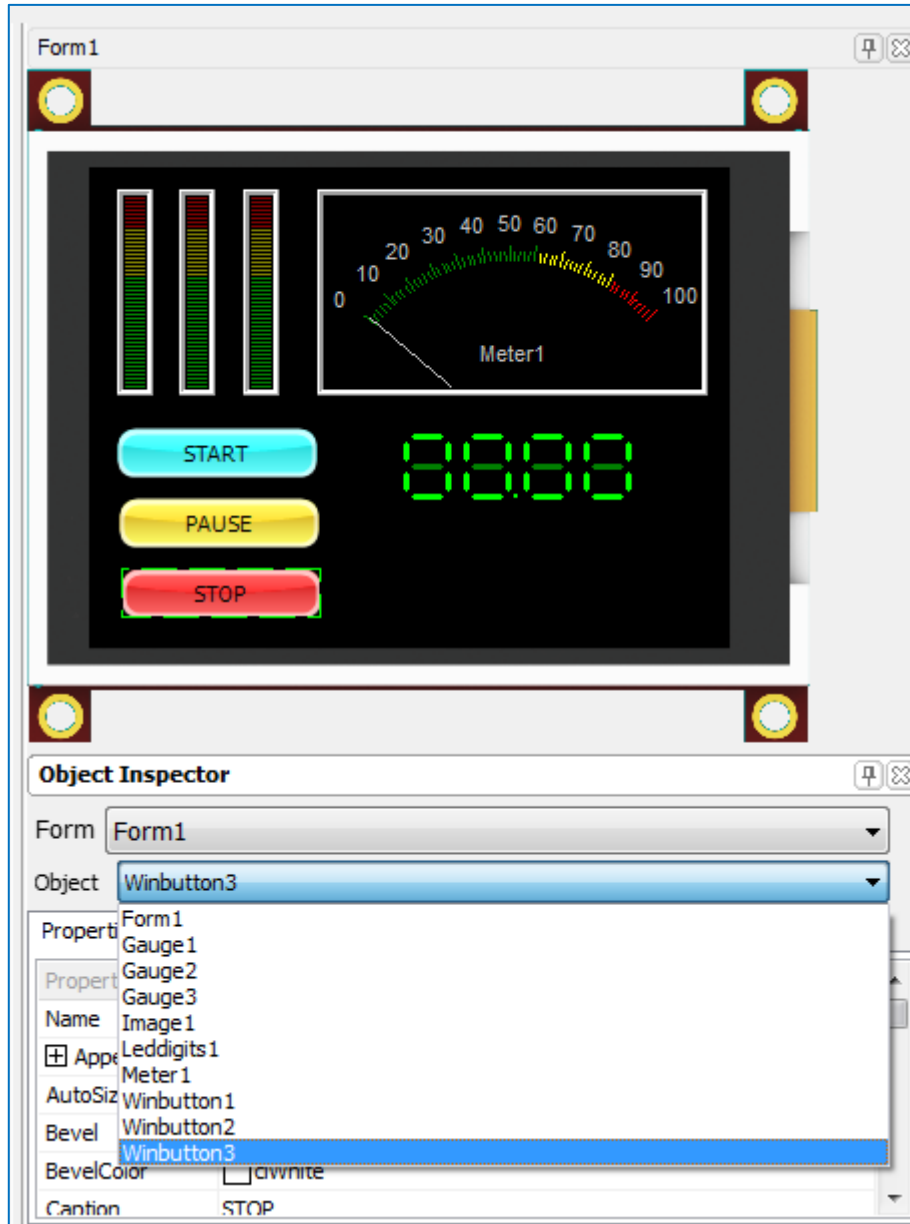
Refer to the next image, which demonstrates this concept.



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.

8.3. Changing Between Object Properties

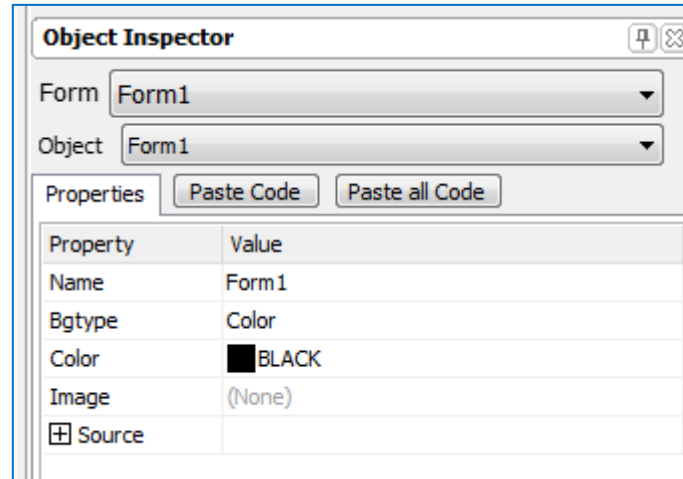
If there are multiple graphical objects on the screen, it may be necessary to change between object properties. Do this by using the dropdown box located at the top of the **Object Inspector**. Alternatively, each object can be edited by clicking on the object in the module display area.



8.4. Inserting a Layout into the User Application Program

When an image layout is finalised, it is ready to have the relevant code behind each object inserted into the user program. From inside the application program, find the desired place to insert the image created using ViSi.

There are two possible options for inserting code into the application.



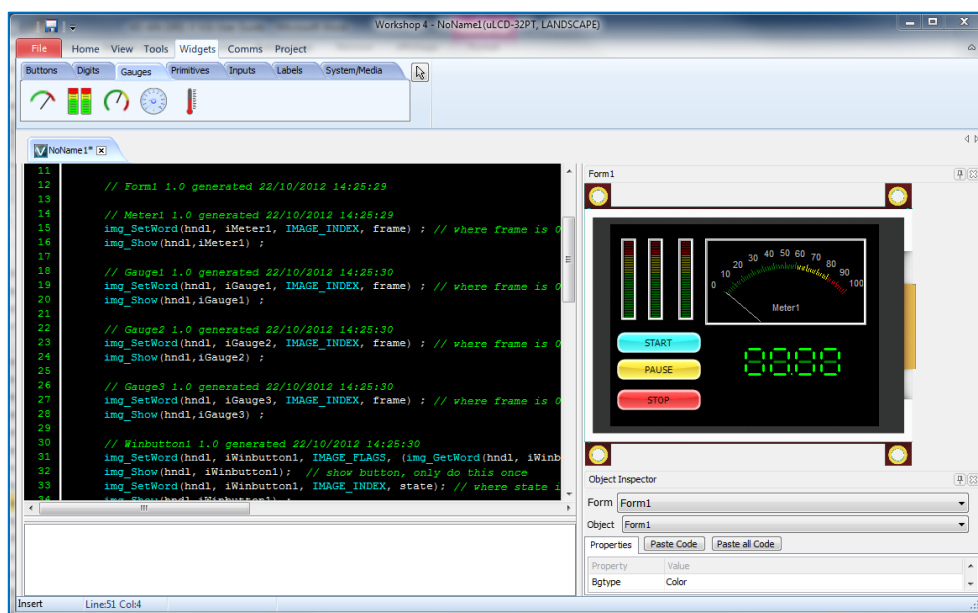
The user can:

- either paste for an individual object;
- or paste all the code in one place for all of the objects created on the screen.

Pasting all of the code can only be done by selecting **Form1**. All other objects only have the ability to generate code for themselves.

The following images illustrate how to paste code into the user application. Select **Form1**, then click **Paste all Code** just below it.

Notice all of the code is inserted into the program:



Often, not all code will be needed in the one place in a given user program. For this reason, it will be necessary to paste code for a particular object only.

The next image demonstrates this point. Select an object –here, **Gauge1**– then click on **Paste Code**. Notice that only a couple of lines of code are inserted into the program.

```
39  
40 // Gauge1 1.0 generated 15/10/2012 16:34:04  
41 img_SetWord(hndl, iGauge1, IMAGE_INDEX, frame) ; // where frame is 0 ;  
42 img_Show(hndl,iGauge1) ;  
43
```

8.5. Finalising a Program for Download

Rounding off a project requires the user to develop the necessary code to control the flow of how the application runs. This will involve writing control loops and sequences in between all of the image screens that have been created using 4D-ViSi.

It is now apparent, that this method of application development has now significantly reduced in time, since all image layouts are generated by 4D-ViSi, which normally the user would have to do.

Not only is time saved in this respect, but also code will no longer be required to be downloaded onto the module every time a visual change is made. Instant graphical changes are now seen in the simulated display, which means an application download will only need to be done once; at the completion of a user program.

8.6. Downloading an Application

When a 4D-ViSi program is compiled, a .gci and .dat file is generated automatically without going through the Graphics Composer software tool. Make sure that the module is plugged into the PC and the COM port is selected. A prompt will appear requiring that the .gci and .dat files are copied onto the micro-SD card before proceeding.

Before clicking OK, Insert the micro-SD card into the PC and locate the .gci and .dat file created during compile.

These will be located in the same folder as the project file. Copy these to the micro-SD card and insert the card back into the module and click OK. The application will finish downloading to the module and the application will now run.

8.7. Example Application Program

The following example was developed by the 4D Team to illustrate some of the features and possibilities with the new 4D-ViSi software tool.

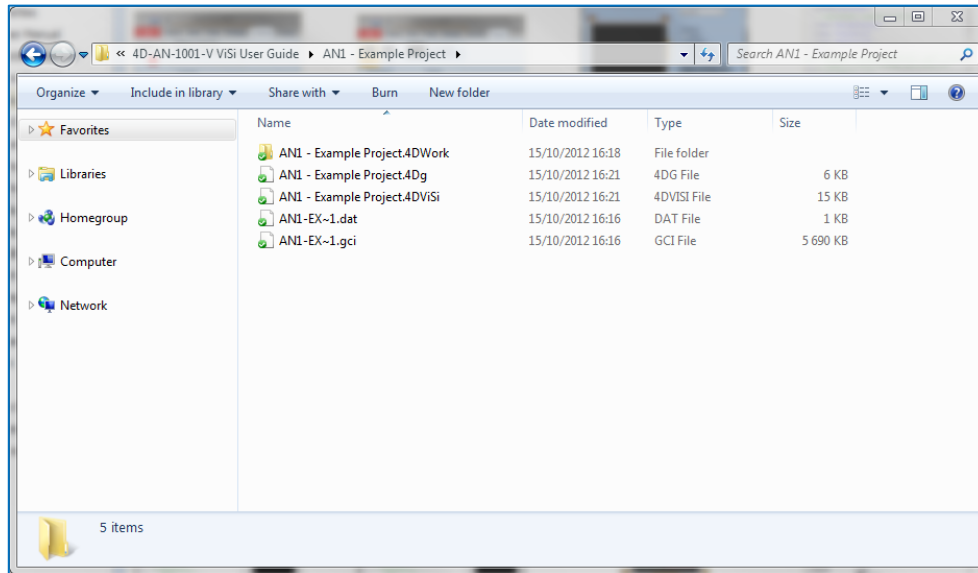
The screen exhibits three gauges, a counter and a meter. All three graphic tools are controlled by the same index and hence all reflect the same value between 1 and 100.

Three buttons control the meters titled; Start, Pause and Stop. The Start button will begin the counter counting from either zero or wherever the counter is up to.

The counter resets every time it reaches 100. The Pause button will stop the counter wherever it is up to; whilst the Stop button will reset the counter to zero.

Examine the code to see just how simple it is to generate a fully graphic and interactive program with only a few lines of code. Comments will be placed throughout the code to explain certain areas in detail.

Place the .4dg and .4DVisi file that comes with this application note into a project folder on the PC. Open the .4dg file through 4D-ViSi to load the example application. Be sure to follow the steps explained in the previous section for loading the .gci file and .dat file onto the micro-SD in order for the program to work.

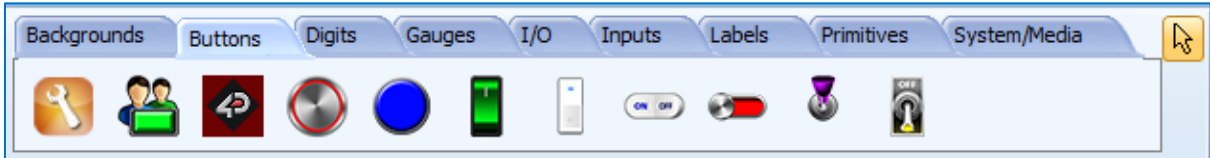


9. Objects

ViSi provides an easy method of designing complex Graphics User Interface applications. The user drags and drops objects on his simulated screen and the code is updated accordingly.

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

9.1. Buttons Object



The Buttons pane contains the following widgets:

- Win Button



- User Button



- Animated Button

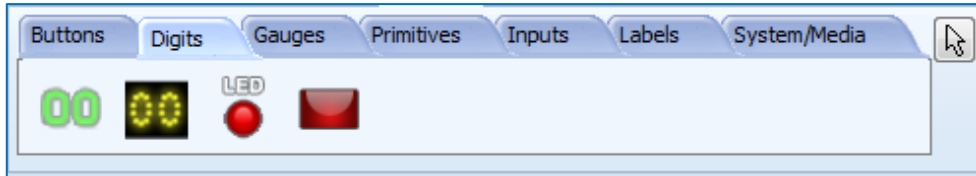


- 4D Buttons



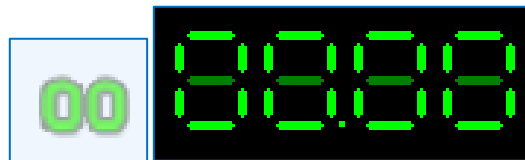


9.2. Digits Objects



The Digits pane contains 4 different displays.

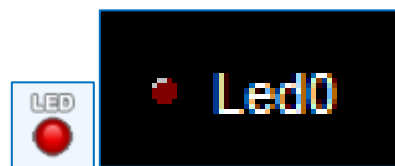
- LED Digits



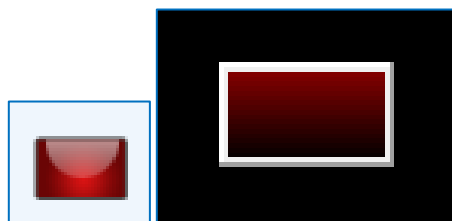
- Custom Digits



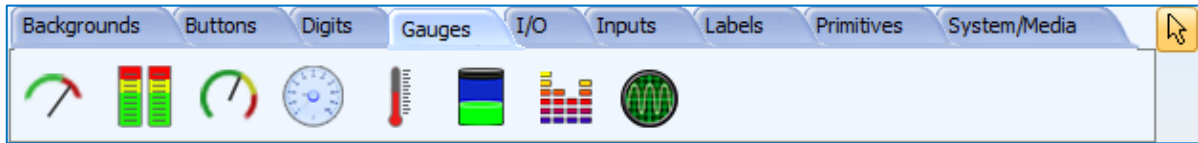
- LED



- User LED

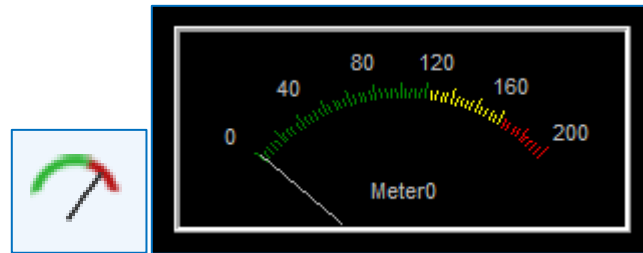


9.3. Gauges Objects

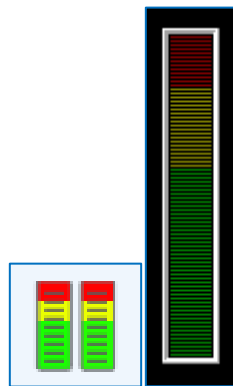


The Gauges pane contains five specialised displays:

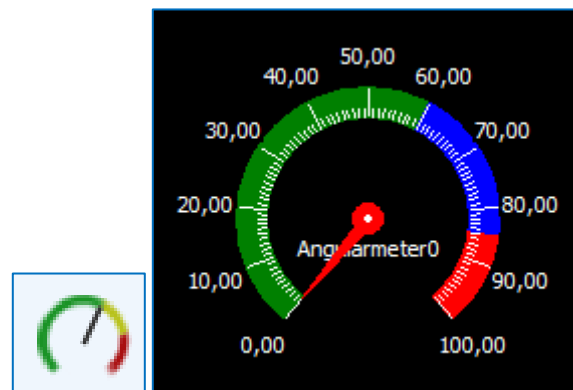
- Meter



- Gauge



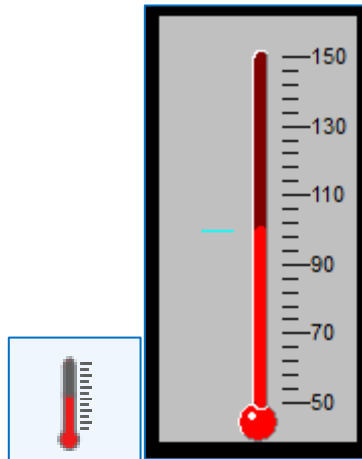
- Angular Meter



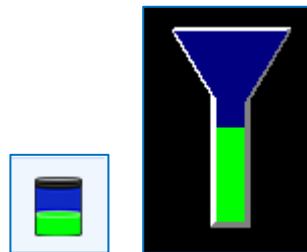
- Cool Gauge



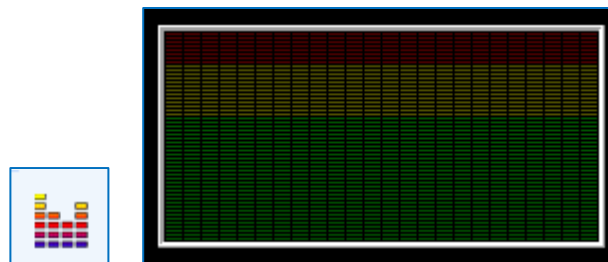
- Thermometer



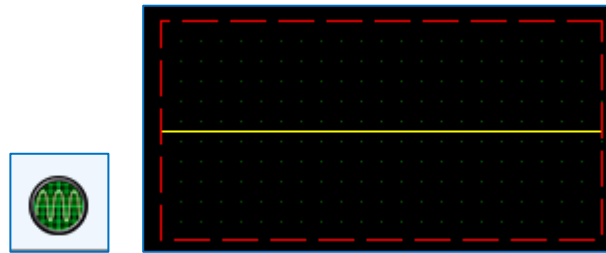
- Tank



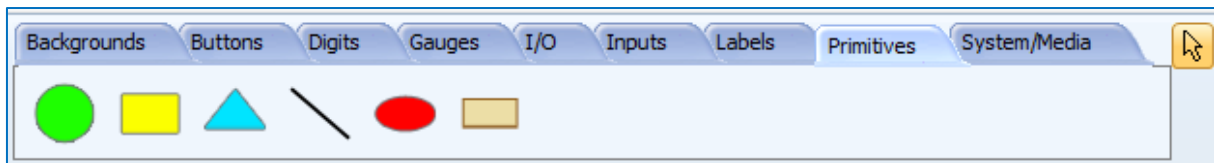
- Spectrum



- Scope

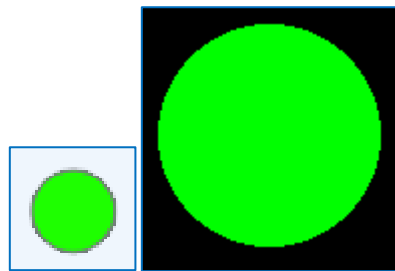


9.4. Primitives Objects

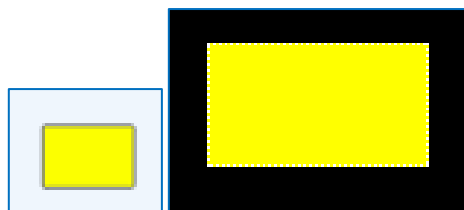


The Primitives pane offers 6 standard static drawings and 2 dynamic objects.

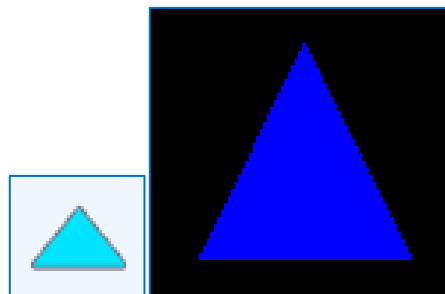
- Circle



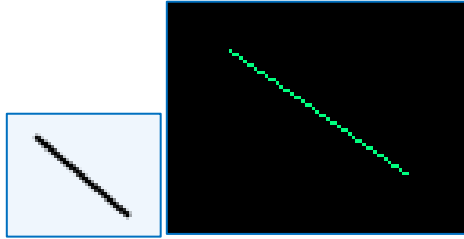
- Rectangle



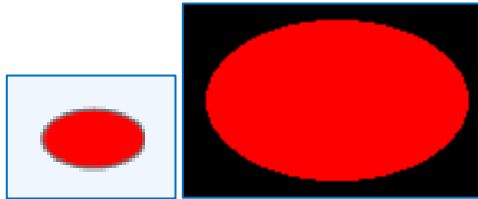
- Triangle



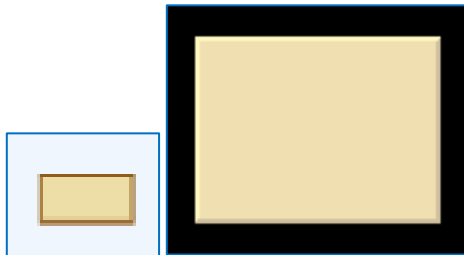
- Line



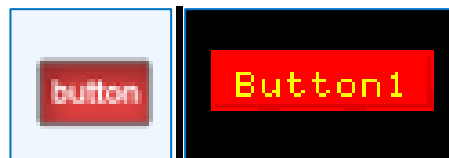
- Ellipse



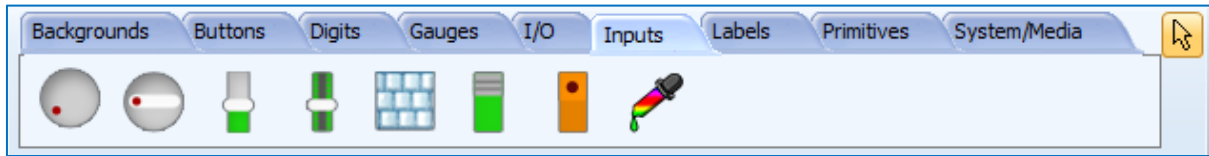
- Panel



- Button

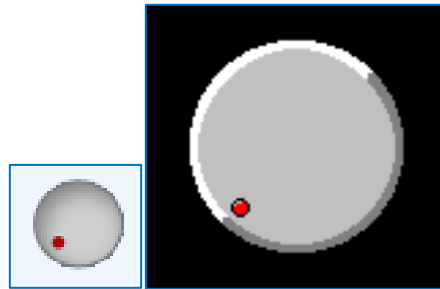


9.5. Inputs Objects



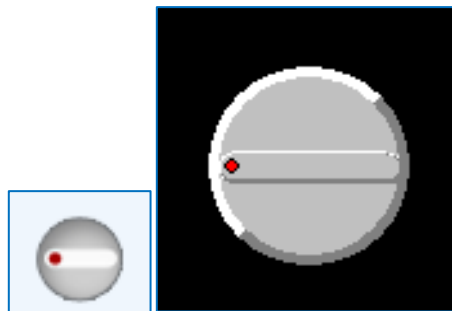
The Inputs pane contains rotary selectors, linear selectors, keyboards and switches.

- Knob



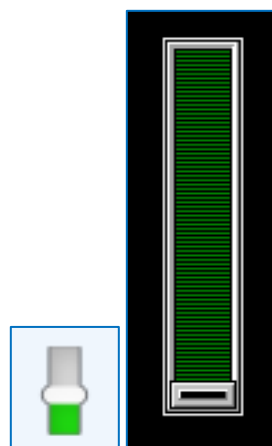
The minimum and maximum angles, the back and the handle can be customised.

- Rotary Switch



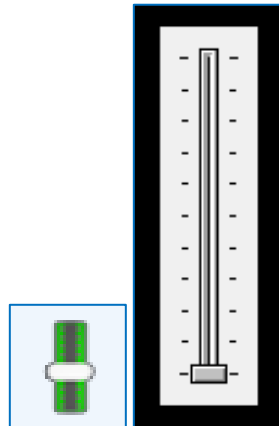
The minimum and maximum angles, the positions and labels, the switch and the winch colours can be customised.

- Slider



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

- Track-bar



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

- Keyboard

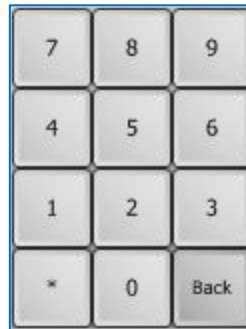


ViSi-Genie comes with various defined keyboards:

- QWERTY keyboard, by default,



- Cell-phone keyboard

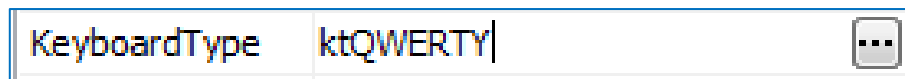


- Numeric keyboard



- And even a customised keyboard.

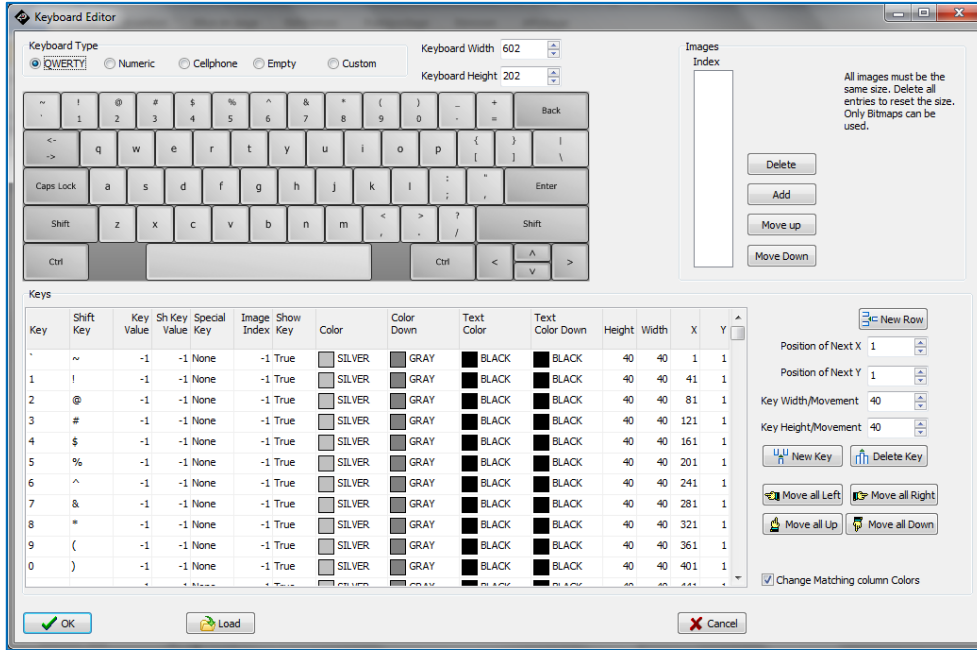
The different keyboards are selected by clicking on the **KeyboardType** property:



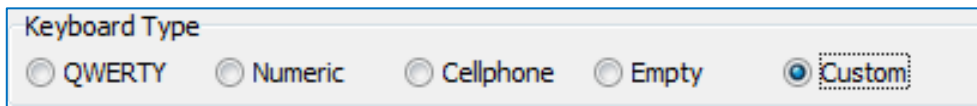
Click on the button



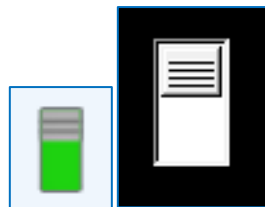
to launch the Keyboard Editor:



The Keyboard Editor allows to select and customise the keyboard:

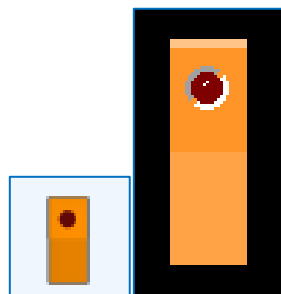


- DIP Switch



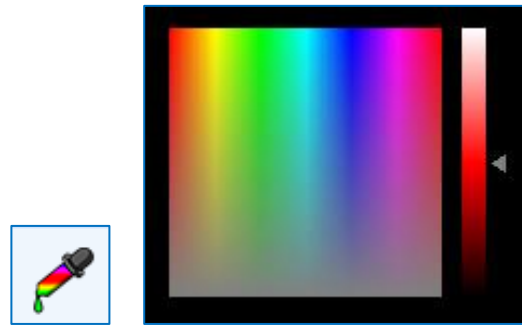
The number of positions of the switch can be specified, 2 as shown or more.

- Rocker Switch

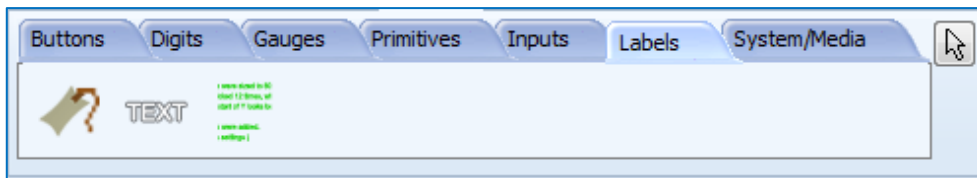


When on, the red LED is turned on.

- Color Picker



9.6. Labels Objects

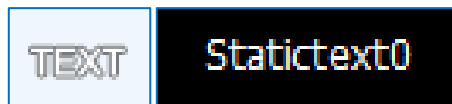


The Labels pane offers three different objects to display text.

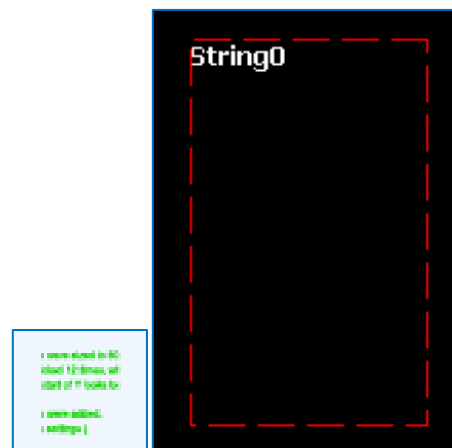
- Label



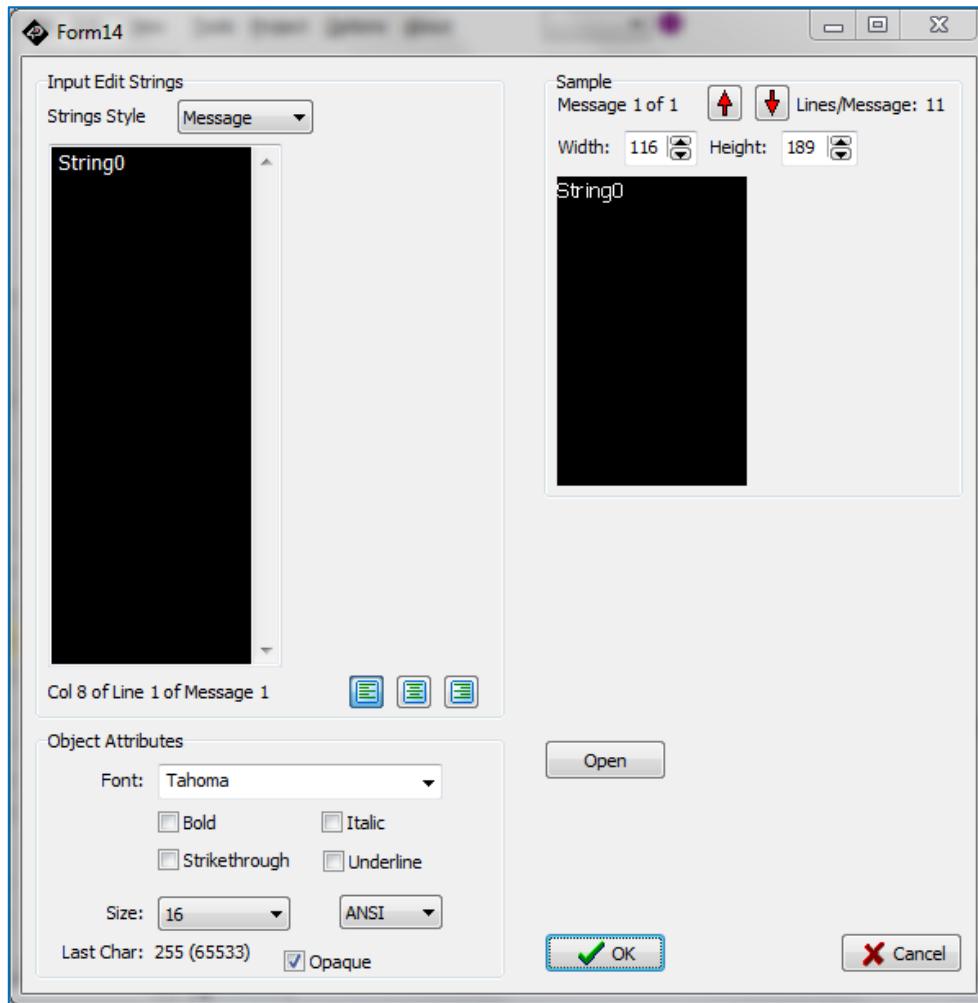
- Static Text



- Strings

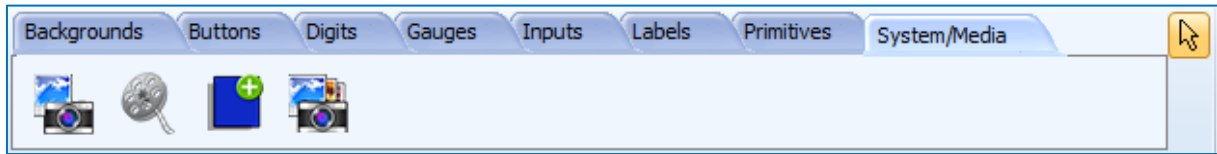


The text is defined by:



Font, size, ANSI or Unicode can be defined.

9.7. System/Media Objects

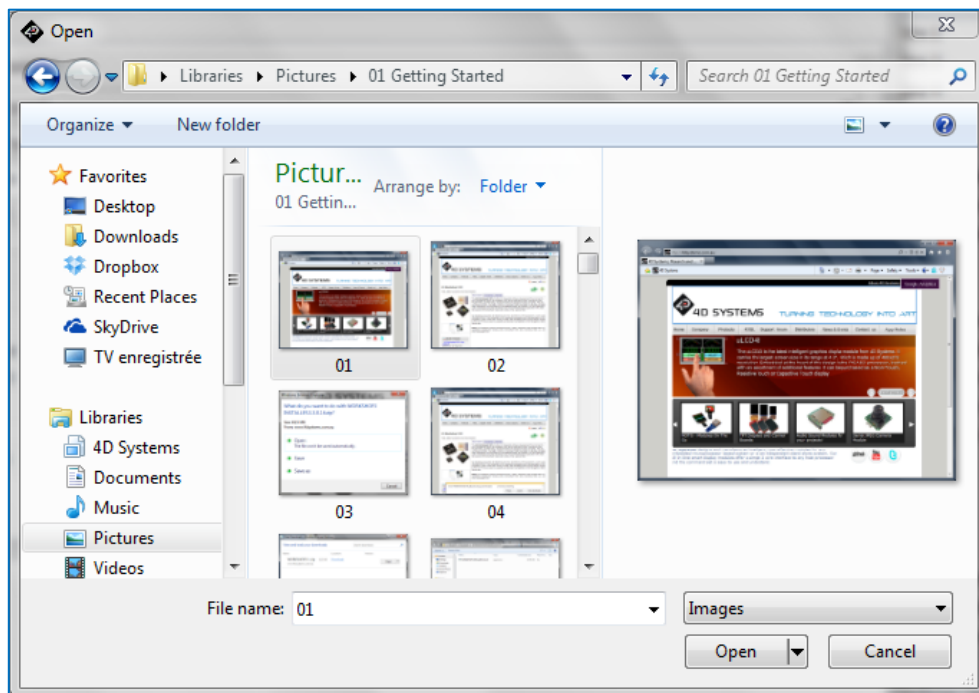


The System pane includes the image, video, form, and user images objects.

- Image



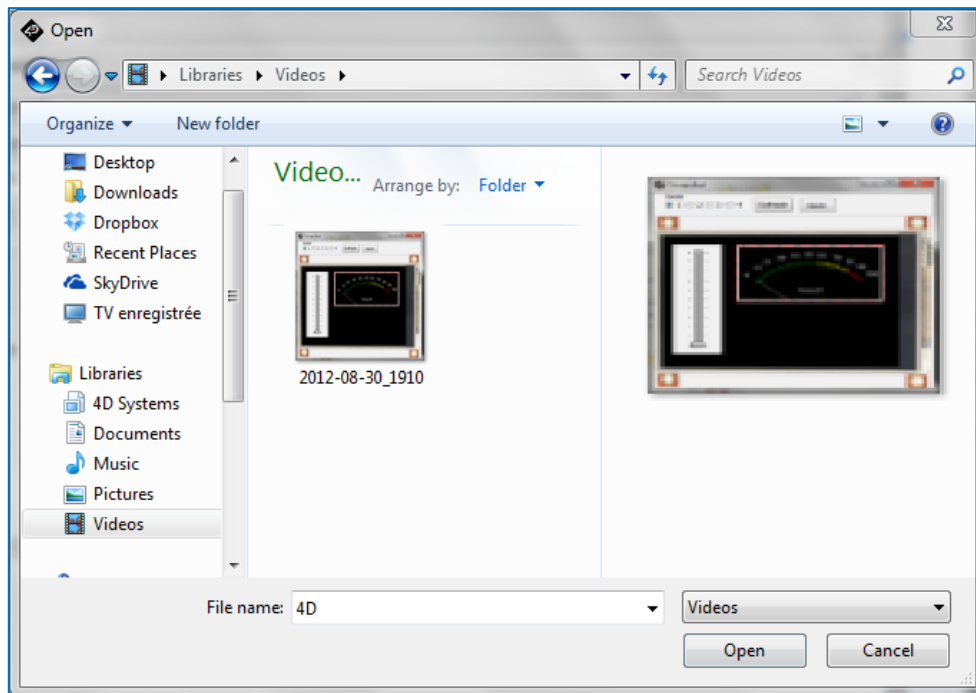
The image is selected through an Open window:



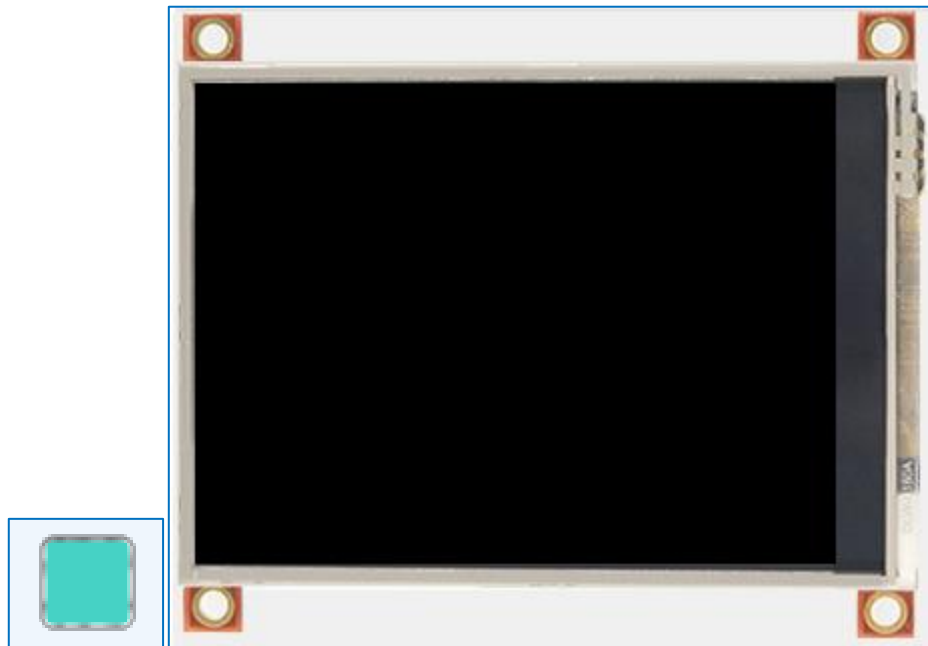
- Video



The video is selected through an Open window:



- Form

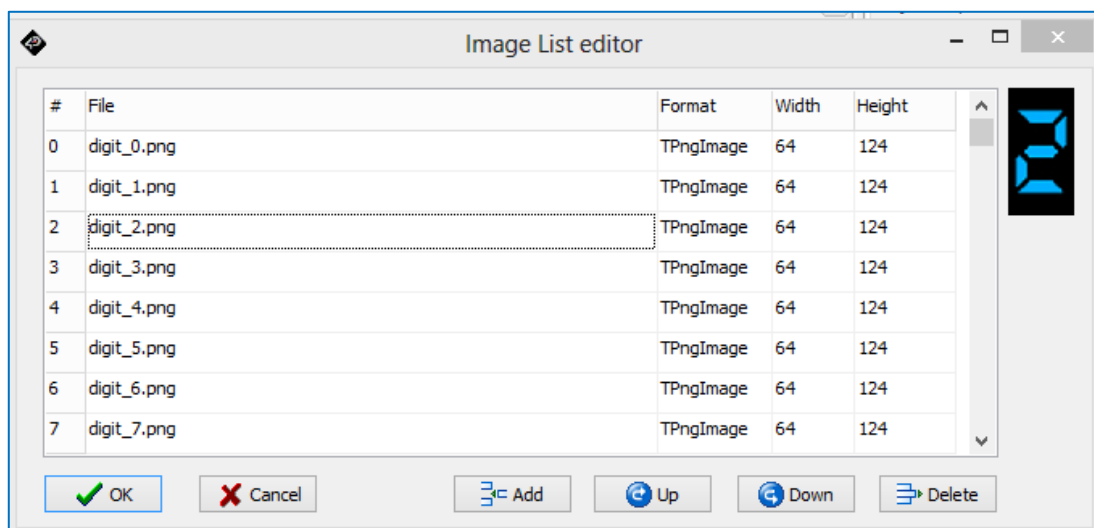


The Form creates a new empty form and adds it to the project.

- 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 images are selected through the Image List editor window.

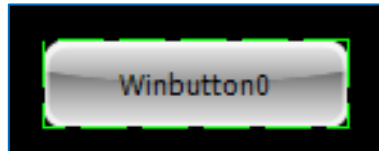


9.8. Selection Tool

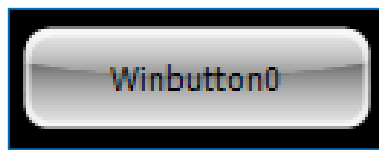


The arrow is used to deselect and object.

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

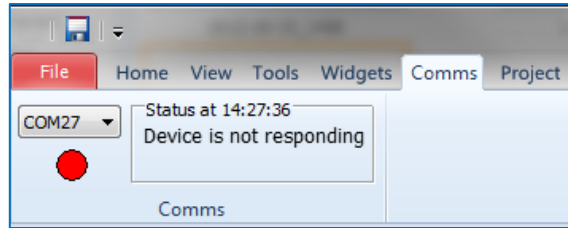


To deselect an object, just click again: the dotted lines disappear.

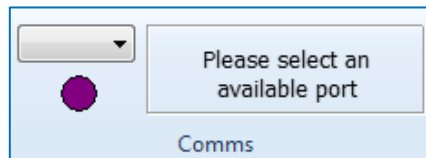


10. Connect the Screen

Connect the screen to a USB port with the 4D Systems programming cable and select the **Comms** menu:

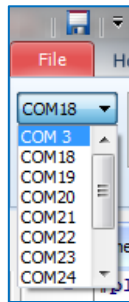


Above the **Comms** section, the **violet** light mentions no screen is currently connected.

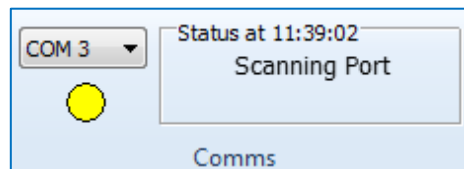


Connect the 4D Systems to the screen and plug the cable into the USB port.

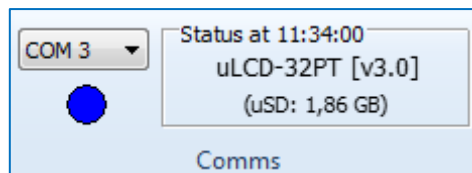
Click on the drop-down list and select the COM port, here COM3.



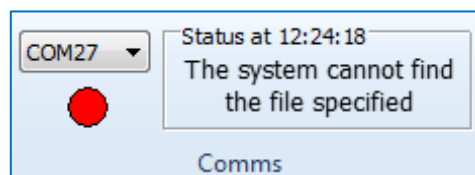
The light turns **yellow** while the connection is being established:



Finally, the light goes **blue** when the connection is established.



The light turns **red** when no screen is attached to the selected port:



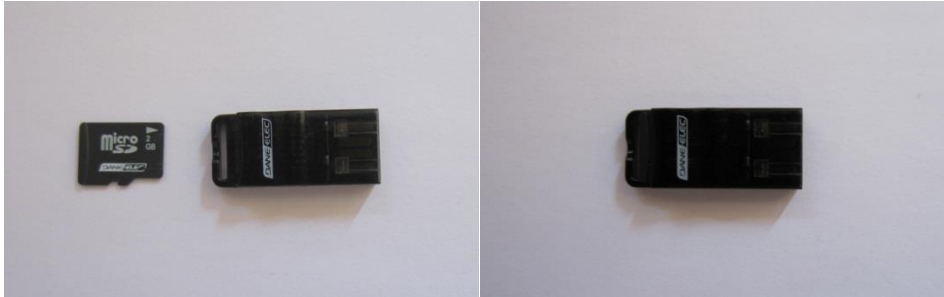
11. Insert the Micro-SD Card

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

For Goldelox modules, the micro-SD card shall not be formatted at all, it requires the SD card to be RAW.

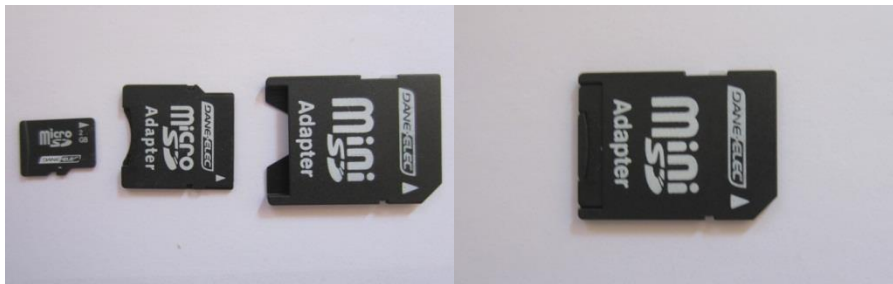
To connect the micro-SD card, either

- Insert the micro-SD card into the USB adaptor and plug the USB adaptor into an USB port of the PC.

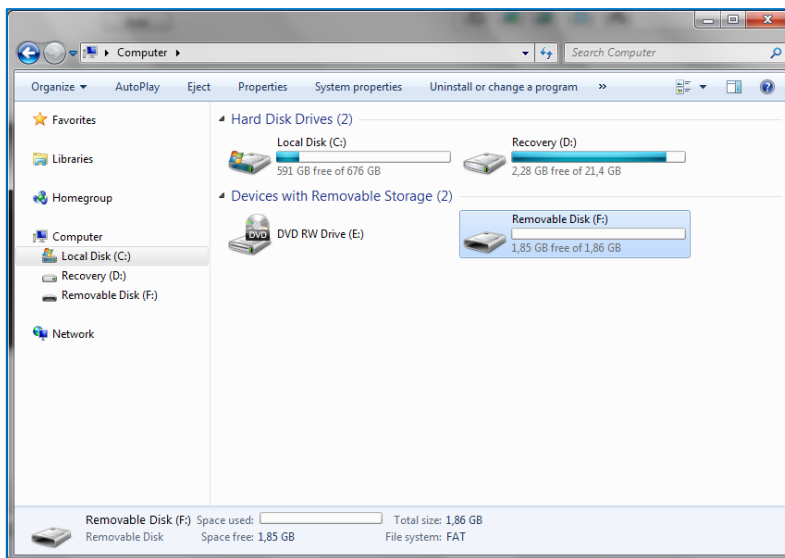


Or

- Insert the micro-SD card into a micro-SD to SD card converter and plug the SD card converter into the SD card slot of the PC.



Check the micro-SD card is mounted, here as drive E:.

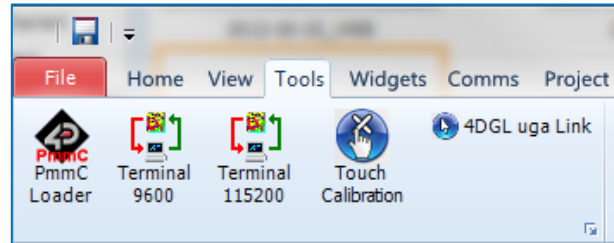


For Goldelox, if prompted to format the SD card, click no/cancel. Leave the card unformatted and Workshop4 will handle the rest.

12. Communication Terminal

An alternative to the debugger is the Terminal.

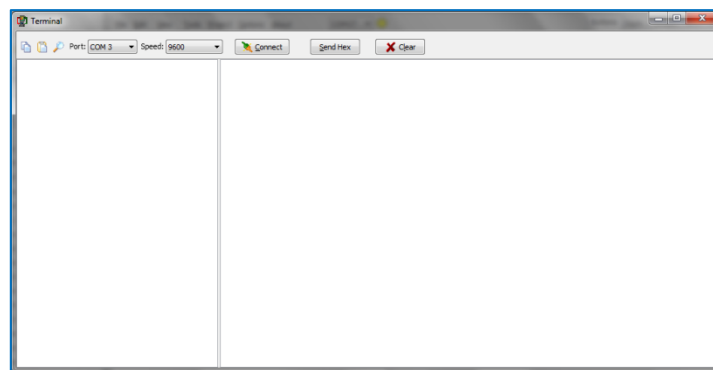
To launch the Terminal, select the **Tools** menu...



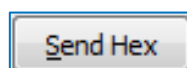
...and

- 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

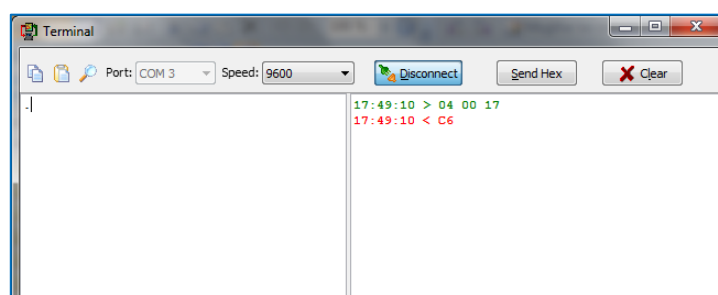


The commands sent by the host and the messages sent by the screen are the same as with the **Genie Test Executor** debugger.

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.



```

#platform "uLCD-32PT"
#inherit "4DGL_16bitColours.fnc"
#inherit "VisualConst.inc"
#inherit "4DViSi 4D-AN-1010Const.inc"
#inherit "ledDigitsDisplay.inc"

var x,y;
var mindex;

func main()
  putstr("Mounting...");
  if (!(disk:=file_Mount()))
    while (!(disk :=file_Mount())) // check micro-SD is loaded
      putstr("Drive not mounted...");
      pause(200);
      gfx_Cls();
      pause(200);
    wend
  endif
  gfx_TransparentColour(0x0020);
  gfx_Transparency(ON);
  hndl := file_LoadImageControl("4DVISI~1.dat", "4DVISI~1.gci", 1);

  img_Show(hndl,iwinbutton1); // meter1 generated 11/8/2011 2:05:10 PM
  img_Show(hndl,iwinbutton2); // winbutton2 generated 11/8/2011 2:05:10 PM
  img_Show(hndl,iwinbutton3); // winbutton3 generated 11/8/2011 2:05:10 PM
  img_Show(hndl, ileddigits1); // leddigits1 generated 11/10/2011 12:54:00 PM
  img_Show(hndl, igauges4); // gauge4 generated 11/10/2011 12:56:25 PM
  img_Show(hndl, igauges1); // gauge1 generated 11/10/2011 4:39:17 PM
  img_Show(hndl, igauges2); // gauge2 generated 11/10/2011 12:59:28 PM

  mindex := 0; // This is the index that controls the value of all the meters

  touch_Set(TOUCH_ENABLE); // enable touch

  repeat
  if(touch_Get(TOUCH_STATUS) == TOUCH_PRESSED) // scan for touch
    x := touch_Get(TOUCH_GETX); // get x coordinate
    y := touch_Get(TOUCH_GETY); // get y coordinate
    if( (x >= 104 && x <= 224) && (y >= 188 && y <= 223) ) // coordinates for Start Button
      img_SetWord(hndl, iwinbutton3, IMAGE_INDEX, 1); // show start button depressed
      img_Show(hndl,iwinbutton3);
      pause(200);
      img_SetWord(hndl, iwinbutton3, IMAGE_INDEX, 0); // release start button
      img_Show(hndl,iwinbutton3);
    repeat
      mindex++; // increment the index
      if( mindex == 100 ) // if the index reaches 100, reset it to 0
        mindex := 0;
      endif
      img_SetWord(hndl, imeter1, IMAGE_INDEX, mindex); // update imeter1
      img_Show(hndl, imeter1);
      ledDigitsDisplay(mindex, ileddigits1+1, 56, 3, 2, 43, 0 ); // update leddigits1
      img_SetWord(hndl, igauges4, IMAGE_INDEX, mindex); // update igauges4
      img_Show(hndl, igauges4);
      img_SetWord(hndl, igauges1, IMAGE_INDEX, mindex); // update ugauges1
      img_Show(hndl, igauges1);
      img_SetWord(hndl, igauges2, IMAGE_INDEX, mindex); // update igauges2
  
```



```

img_Show(hndl, igauge2);
if(touch_Get(TOUCH_STATUS) == TOUCH_PRESSED)
  x := touch_Get(TOUCH_GETX);
  y := touch_Get(TOUCH_GETY);
  if((x >= 104 && x <= 224) && (y >= 228 && y <= 263)) || ((x >= 104 && x <= 224) && (y >= 268
&& y <= 303)))
    break; // if a touch is detected on either of the other buttons, break out of this forever loop
  endif
endif
forever
endif
if( (x >= 104 && x <= 224) && (y >= 228 && y <= 263) ) // coordinates for Pause Button
  img_SetWord(hndl, iwinbutton2, IMAGE_INDEX, 1); // show pause button depressed
  img_Show(hndl, iwinbutton2);
  pause(200);
  img_SetWord(hndl, iwinbutton2, IMAGE_INDEX, 0); // release pause button
  img_Show(hndl, iwinbutton2);
  img_SetWord(hndl, imeter1, IMAGE_INDEX, mindex); // update imeter1
  img_Show(hndl, imeter1);
  ledDigitsDisplay(mindex, ileddigits1+1, 56, 3, 2, 43, 0); // update leddigits1
  img_SetWord(hndl, igauge4, IMAGE_INDEX, mindex); // update igauge4
  img_Show(hndl, igauge4);
  img_SetWord(hndl, igauge1, IMAGE_INDEX, mindex); // update igauge1
  img_Show(hndl, igauge1);
  img_SetWord(hndl, igauge2, IMAGE_INDEX, mindex); // update igauge2
  img_Show(hndl, igauge2);
endif
if( (x >= 104 && x <= 224) && (y >= 268 && y <= 303) ) // coordinates for Stop Button
  img_SetWord(hndl, iwinbutton1, IMAGE_INDEX, 1); // show stop button depressed
  img_Show(hndl, iwinbutton1);
  pause(200);
  img_SetWord(hndl, iwinbutton1, IMAGE_INDEX, 0); // release stop button
  img_Show(hndl, iwinbutton1);
  mindex := 0; // reset the index
  img_SetWord(hndl, imeter1, IMAGE_INDEX, mindex); // update imeter1
  img_Show(hndl, imeter1);
  ledDigitsDisplay(mindex, ileddigits1+1, 56, 3, 2, 43, 0); // update leddigits1
  img_SetWord(hndl, igauge4, IMAGE_INDEX, mindex); // update igauge4
  img_Show(hndl, igauge4);
  img_SetWord(hndl, igauge1, IMAGE_INDEX, mindex); // update igauge1
  img_Show(hndl, igauge1);
  img_SetWord(hndl, igauge2, IMAGE_INDEX, mindex); // update igauge2
  img_Show(hndl, igauge2);
endif
endif

forever
endfunc

```

13. Application Notes

For a more detailed presentation of the objects with examples, please refer to the corresponding application notes:

Reference	Content
4D-AN-P3002	Adding and Configuring a DIP Switch
4D-AN-P3003	GPIO Bus Control Using DIP Switch Objects
4D-AN-P3004	Adding and Configuring Fancy Buttons
4D-AN-P3005	Adding Image and Video
4D-AN-P4001	Getting Started - First Project with ViSi-Genie
4D-AN-P4002	ViSi-Genie - onChanging and onChanged Events
4D-AN-P4003	ViSi-Genie - Customised Keyboard
4D-AN-P4004	ViSi-Genie - Advanced Buttons
4D-AN-P4005	ViSi-Genie - Show Image
4D-AN-P4006	ViSi-Genie - Play Sound
4D-AN-P4007	ViSi-Genie - Play Video
4D-AN-P4008	ViSi-Genie - Gauges
4D-AN-P4009	ViSi-Genie - Inputs
4D-AN-P4010	ViSi-Genie - Connection to an Arduino Host with Red-Green-Blue LED Control
4D-AN-P4011	ViSi-Genie - Using Combined Objects
4D-AN-P4012	ViSi-Genie - Digital Displays
4D-AN-P4013	ViSi-Genie - Labels, Texts and Strings
4D-AN-D4002	ViSi-Genie - Tank
4D-AN-D4003	ViSi-Genie - Spectrum
4D-AN-D4004	ViSi-Genie - Single Trace Scope
4D-AN-D4005	ViSi-Genie - Color Picker
4D-AN-D4006	ViSi-Genie - User Button
4D-AN-D4007	ViSi-Genie - Animated Button
4D-AN-D4008	ViSi-Genie - 4D Buttons
4D-AN-D4009	ViSi-Genie - Pin Input and Output

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

14. Revision History

Revision	Revision Content	Revision Date
1.0	First Release	Nov 16, 2012
1.1	Amended microSD detail	July 7, 2013
1.2	Added the new widgets	Sept 23, 2013
1.3	Added Z-order description to Section 8.2	Dec 22, 2014

15. Legal Notice

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. 4D Systems 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 Systems 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 Systems range of products, however the quality may vary.

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.

16. Contact Information

For Technical Support: support@4dsystems.com.au

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

Website: www.4dsystems.com.au

Copyright 4D Systems Pty. Ltd. 2000-2013.