4D LABS SEMICONDUCTORS

Workshop4

Integrated Development Environment

USER GUIDE

Document Revision: 2.3

Document Date: 27th July 2020

Table of Contents

1. Description

This user guide provides an introduction to Workshop4, the 4D integrated development environment. Workshop4 supports multiple development environments for the user, to cater for different user requirements and skill level.

The **Designer** environment enables the user to write 4DGL code in its natural form to program the 4D processor/module of choice.

A visual programming experience, suitably called **ViSi**, enables drag-and-drop type placement of objects to assist with 4DGL code generation and allows the user to visualise how the display will look while being developed.

An advanced environment called **ViSi-Genie** doesn't require any 4DGL coding at all (PRO however enables 4DGL code for a more powerful user interface), it is all done automatically. Simply lay the display out with the objects required, set the events to drive them and the code is written for the user automatically. ViSi-Genie provides the latest rapid development experience from 4D. (Not available for Goldelox)

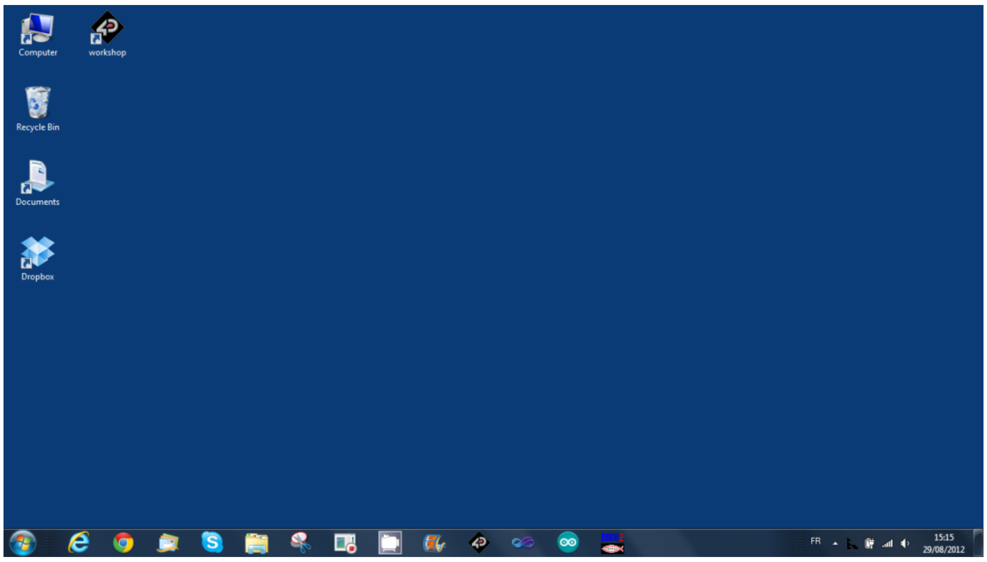
A **Serial** environment is also provided to transform display modules powered by 4D Labs processors into a slave serial module, allowing the user to control the display from any host microcontroller or device with a serial port.

Additionally, Workshop4 also offers Arduino compatible environments that allows the user to easily create a project with both a 4D and an Arduino product. More details can be found in Choose Your Arduino Compatible Environment.

To install Workshop4, please refer to the document '***Workshop4 Installation***'.

2. Workshop4

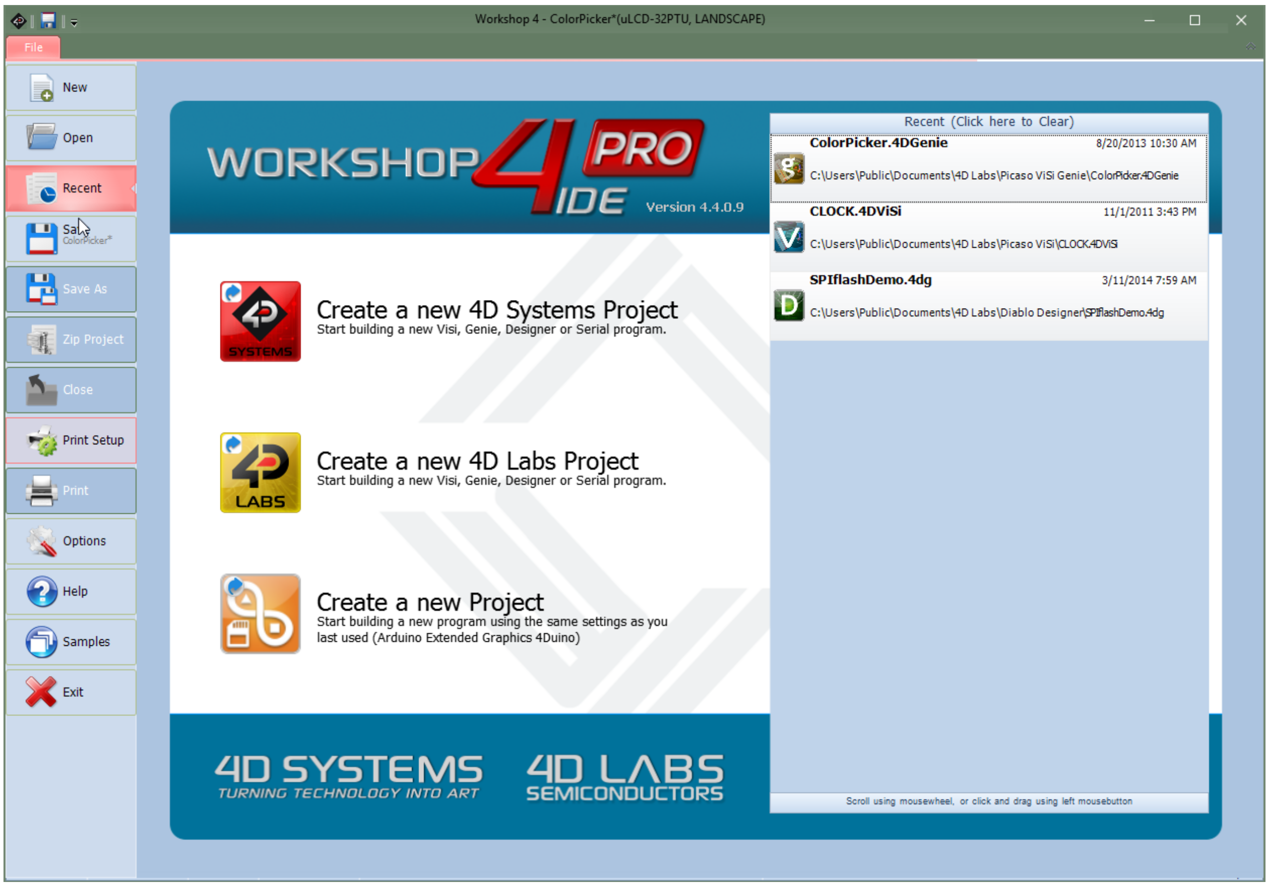
There is an alias for 4D Workshop on the desktop:



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

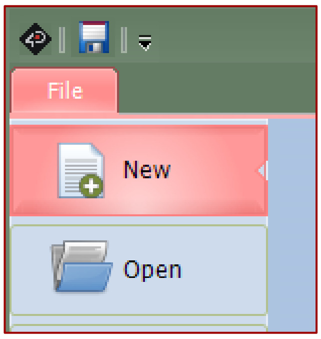
3. Create a New Project

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

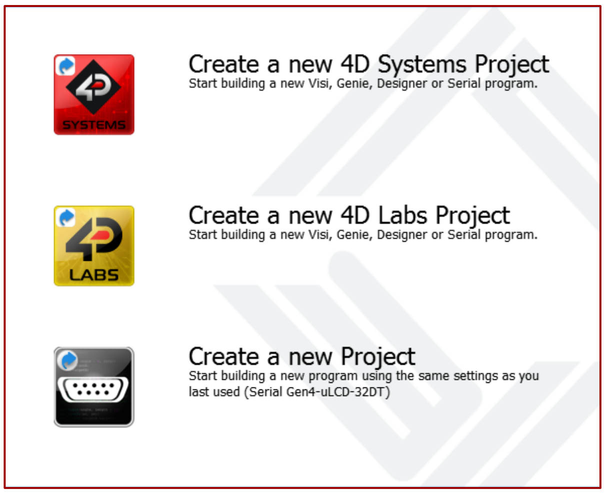

Workshop 4 - ColorPicker*(uLCD-32PTU, LANDSCAPE)
File
New
Open
Recent
Save
Save As
Zip Project
Close
Print Setup
Print
Options
Help
Samples
Exit
WORKSHOP 4 PRO IDE Version 4.4.0.9
Create a new 4D Systems Project
Start building a new Visi, Genie, Designer or Serial program.
Create a new 4D Labs Project
Start building a new Visi, Genie, Designer or Serial program.
Create a new Project
Start building a new program using the same settings as you last used (Arduino Extended Graphics 4Duino)
4D SYSTEMS TURNING TECHNOLOGY INTO ART
4D LABS SEMICONDUCTORS
Recent (Click here to Clear)
ColorPicker.4DGenie 8/20/2013 10:30 AM
C:\Users\Public\Documents\4D Labs\Picaso ViSi Genie\ColorPicker.4DGenie
CLOCK.4DViSi 11/1/2011 3:43 PM
C:\Users\Public\Documents\4D Labs\Picaso ViSi\CLOCK.4DViSi
SPIflashDemo.4dg 3/11/2014 7:59 AM
C:\Users\Public\Documents\4D Labs\Diablo Designer\SPIflashDemo.4dg
Scroll using mousewheel, or click and drag using left mousebutton

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 4D Systems Project** on top
- Click on the **Create a new Project** to create a project instance based on your last project settings.



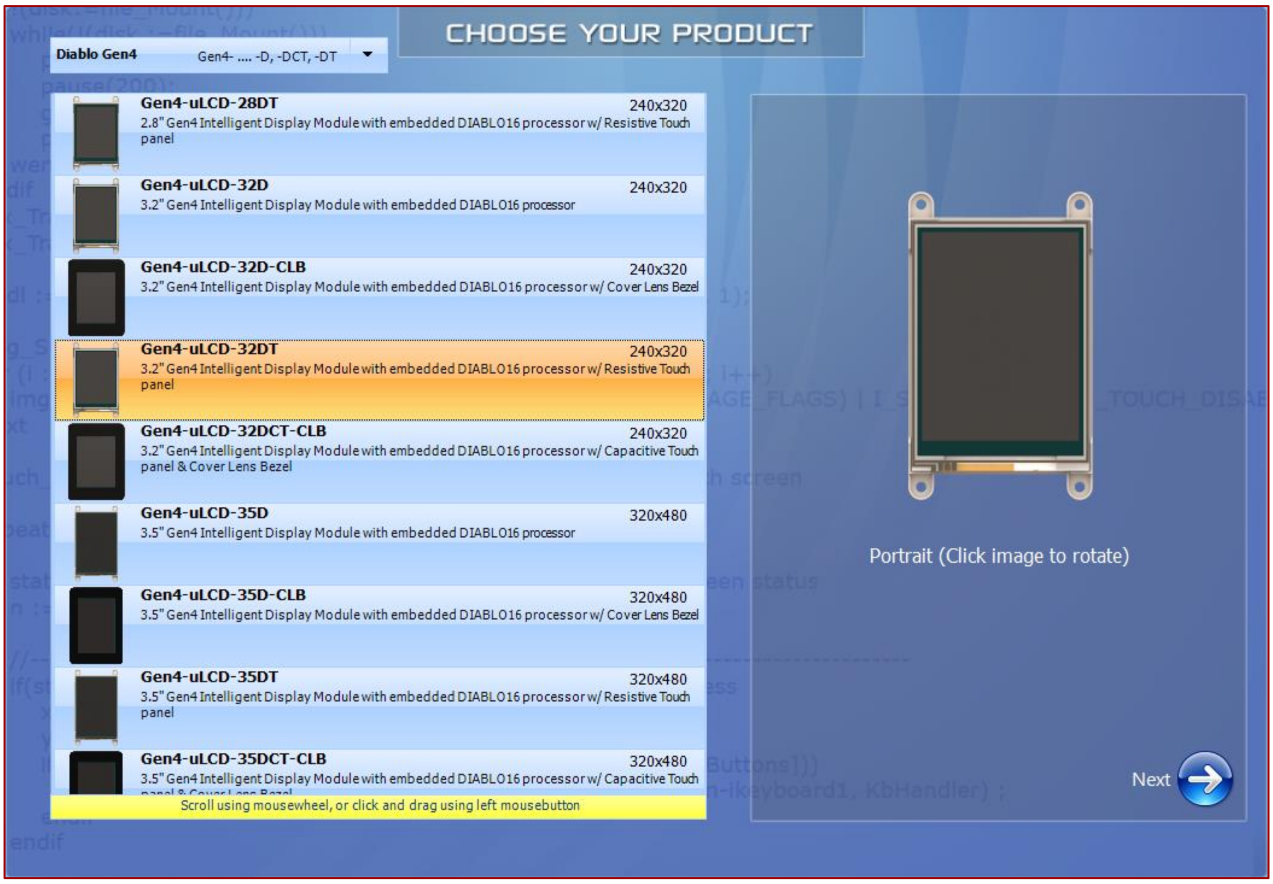
Create a new 4D Systems Project
Start building a new Visi, Genie, Designer or Serial program.

Create a new 4D Labs Project
Start building a new Visi, Genie, Designer or Serial program.

Create a new Project
Start building a new program using the same settings as you last used (Serial Gen4-uLCD-32DT)

Note: Another option is displayed on the image above. **Create a new 4D Labs Project** is an upcoming Workshop4 feature which allows to create a project with a customized display module powered by 4D Labs' processors

Both of the first two options update the main window with the selection of the screen:



CHOOSE YOUR PRODUCT

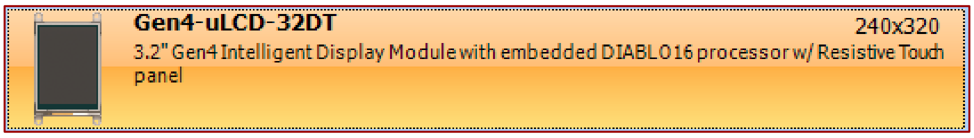
Diablo Gen4 — Gen4- .... -D, -DCT, -DT

Gen4-uLCD-28DT 2.8" Gen4 Intelligent Display Module with embedded DIABLO16 processor w/ Resistive Touch panel	240x320
Gen4-uLCD-32D 3.2" Gen4 Intelligent Display Module with embedded DIABLO16 processor	240x320
Gen4-uLCD-32D-CLB 3.2" Gen4 Intelligent Display Module with embedded DIABLO16 processor w/ Cover Lens Bezel	240x320
Gen4-uLCD-32DT 3.2" Gen4 Intelligent Display Module with embedded DIABLO16 processor w/ Resistive Touch panel	240x320
Gen4-uLCD-32DCT-CLB 3.2" Gen4 Intelligent Display Module with embedded DIABLO16 processor w/ Capacitive Touch panel & Cover Lens Bezel	240x320
Gen4-uLCD-35D 3.5" Gen4 Intelligent Display Module with embedded DIABLO16 processor	320x480
Gen4-uLCD-35D-CLB 3.5" Gen4 Intelligent Display Module with embedded DIABLO16 processor w/ Cover Lens Bezel	320x480
Gen4-uLCD-35DT 3.5" Gen4 Intelligent Display Module with embedded DIABLO16 processor w/ Resistive Touch panel	320x480
Gen4-uLCD-35DCT-CLB 3.5" Gen4 Intelligent Display Module with embedded DIABLO16 processor w/ Capacitive Touch panel & Cover Lens Bezel	320x480

Scroll using mousewheel, or click and drag using left mousebutton

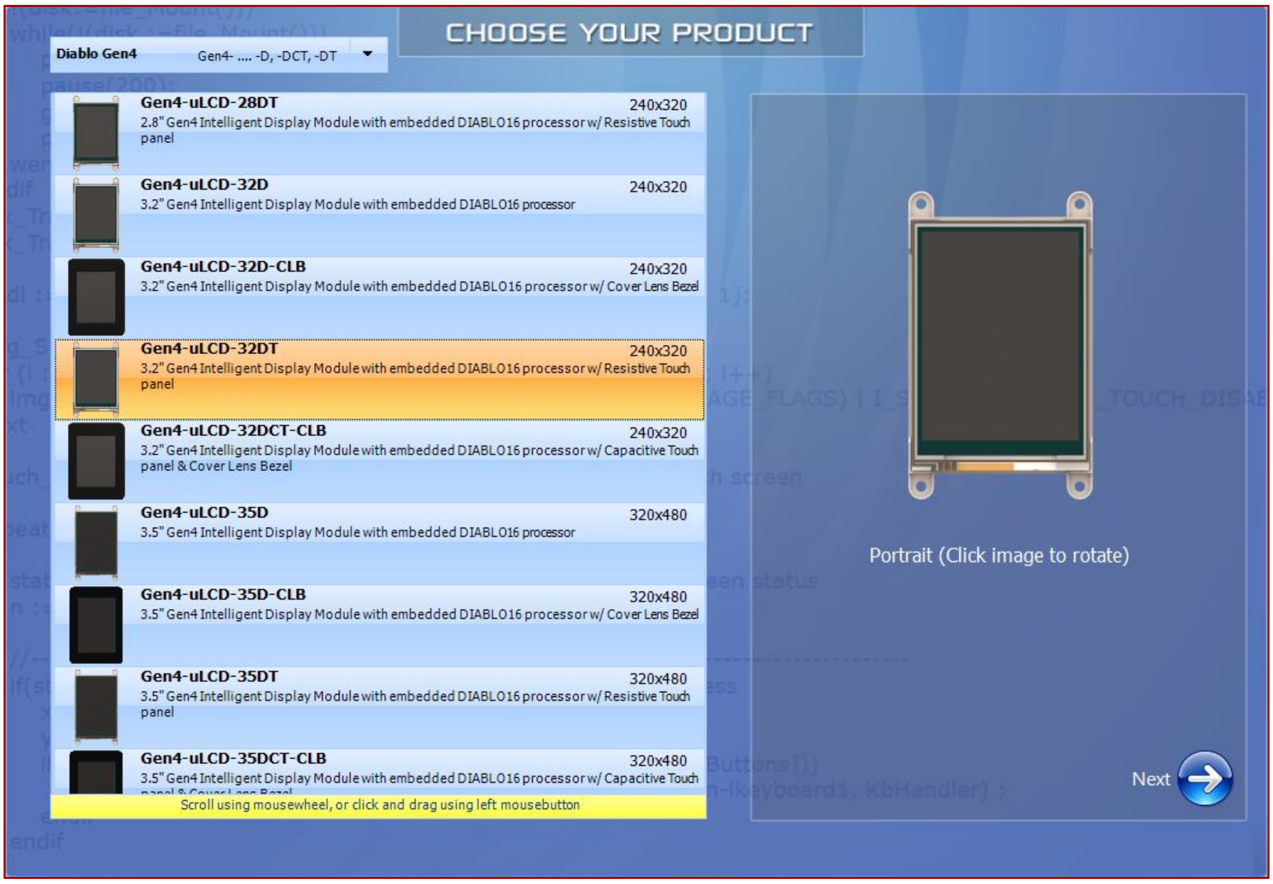
Portrait (Click image to rotate)

Select the screen, here the gen4-uLCD-32DT



Gen4-uLCD-32DT 240x320
3.2" Gen4 Intelligent Display Module with embedded DIABLO16 processor w/ Resistive Touch panel

The selected screen is displayed:



CHOOSE YOUR PRODUCT
Diablo Gen4
Gen4- .... -D, -DCT, -DT
Portrait (Click image to rotate)
Next
Scroll using mousewheel, or click and drag using left mousebutton

Gen4-uLCD-28DT 240x320
2.8" Gen4 Intelligent Display Module with embedded DIABLO16 processor w/ Resistive Touch panel

Gen4-uLCD-32D 240x320
3.2" Gen4 Intelligent Display Module with embedded DIABLO16 processor

Gen4-uLCD-32D-CLB 240x320
3.2" Gen4 Intelligent Display Module with embedded DIABLO16 processor w/ Cover Lens Bezel

Gen4-uLCD-32DT 240x320
3.2" Gen4 Intelligent Display Module with embedded DIABLO16 processor w/ Resistive Touch panel

Gen4-uLCD-32DCT-CLB 240x320
3.2" Gen4 Intelligent Display Module with embedded DIABLO16 processor w/ Capacitive Touch panel & Cover Lens Bezel

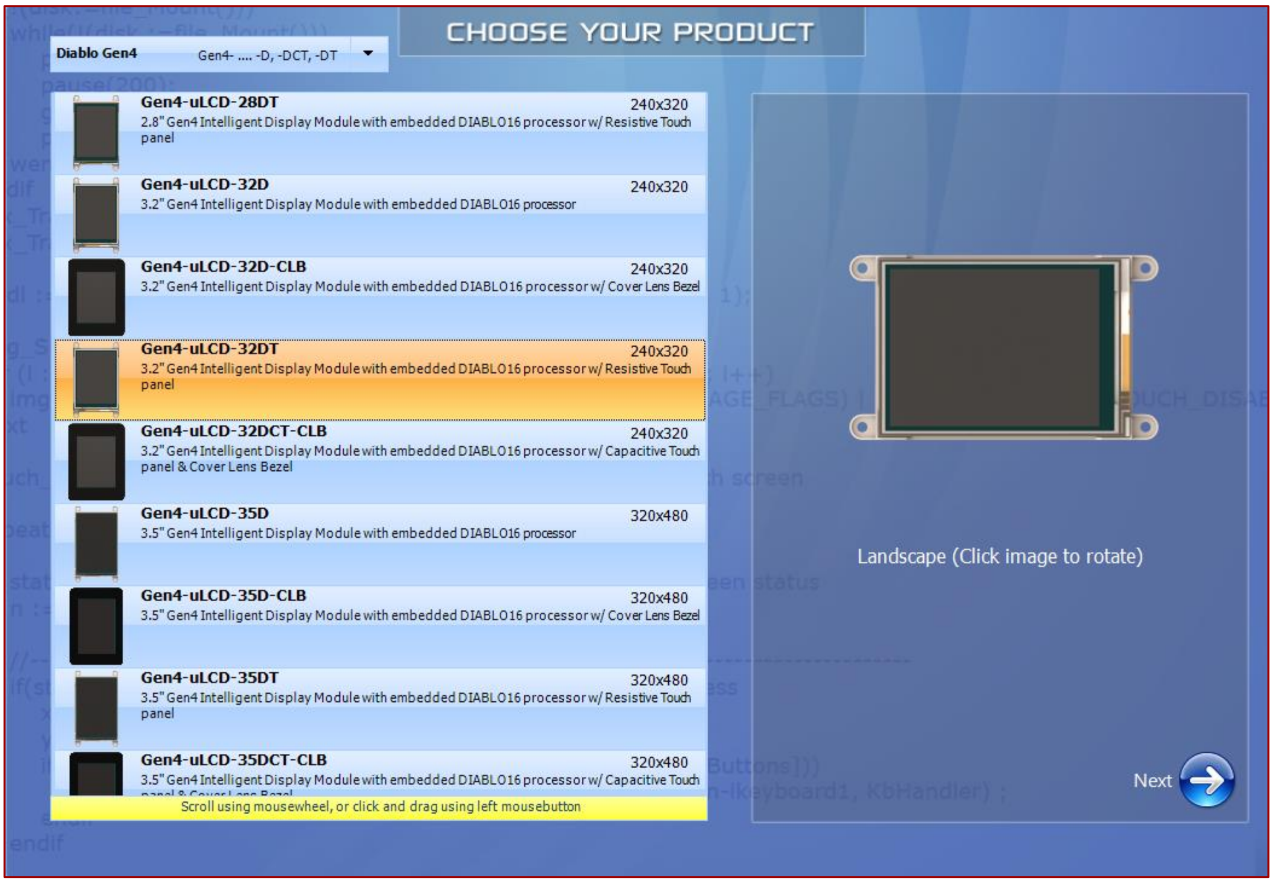
Gen4-uLCD-35D 320x480
3.5" Gen4 Intelligent Display Module with embedded DIABLO16 processor

Gen4-uLCD-35D-CLB 320x480
3.5" Gen4 Intelligent Display Module with embedded DIABLO16 processor w/ Cover Lens Bezel

Gen4-uLCD-35DT 320x480
3.5" Gen4 Intelligent Display Module with embedded DIABLO16 processor w/ Resistive Touch panel

Gen4-uLCD-35DCT-CLB 320x480
3.5" Gen4 Intelligent Display Module with embedded DIABLO16 processor w/ Capacitive Touch

Orientation is portrait by default. To set it to landscape, just click on the image of the screen to rotate it:


CHOOSE YOUR PRODUCT
Diablo Gen4
Gen4- .... -D, -DCT, -DT
Landscape (Click image to rotate)
Next
Scroll using mousewheel, or click and drag using left mousebutton

Gen4-uLCD-28DT 240x320
2.8" Gen4 Intelligent Display Module with embedded DIABLO16 processor w/ Resistive Touch panel

Gen4-uLCD-32D 240x320
3.2" Gen4 Intelligent Display Module with embedded DIABLO16 processor

Gen4-uLCD-32D-CLB 240x320
3.2" Gen4 Intelligent Display Module with embedded DIABLO16 processor w/ Cover Lens Bezel

Gen4-uLCD-32DT 240x320
3.2" Gen4 Intelligent Display Module with embedded DIABLO16 processor w/ Resistive Touch panel

Gen4-uLCD-32DCT-CLB 240x320
3.2" Gen4 Intelligent Display Module with embedded DIABLO16 processor w/ Capacitive Touch panel & Cover Lens Bezel

Gen4-uLCD-35D 320x480
3.5" Gen4 Intelligent Display Module with embedded DIABLO16 processor

Gen4-uLCD-35D-CLB 320x480
3.5" Gen4 Intelligent Display Module with embedded DIABLO16 processor w/ Cover Lens Bezel

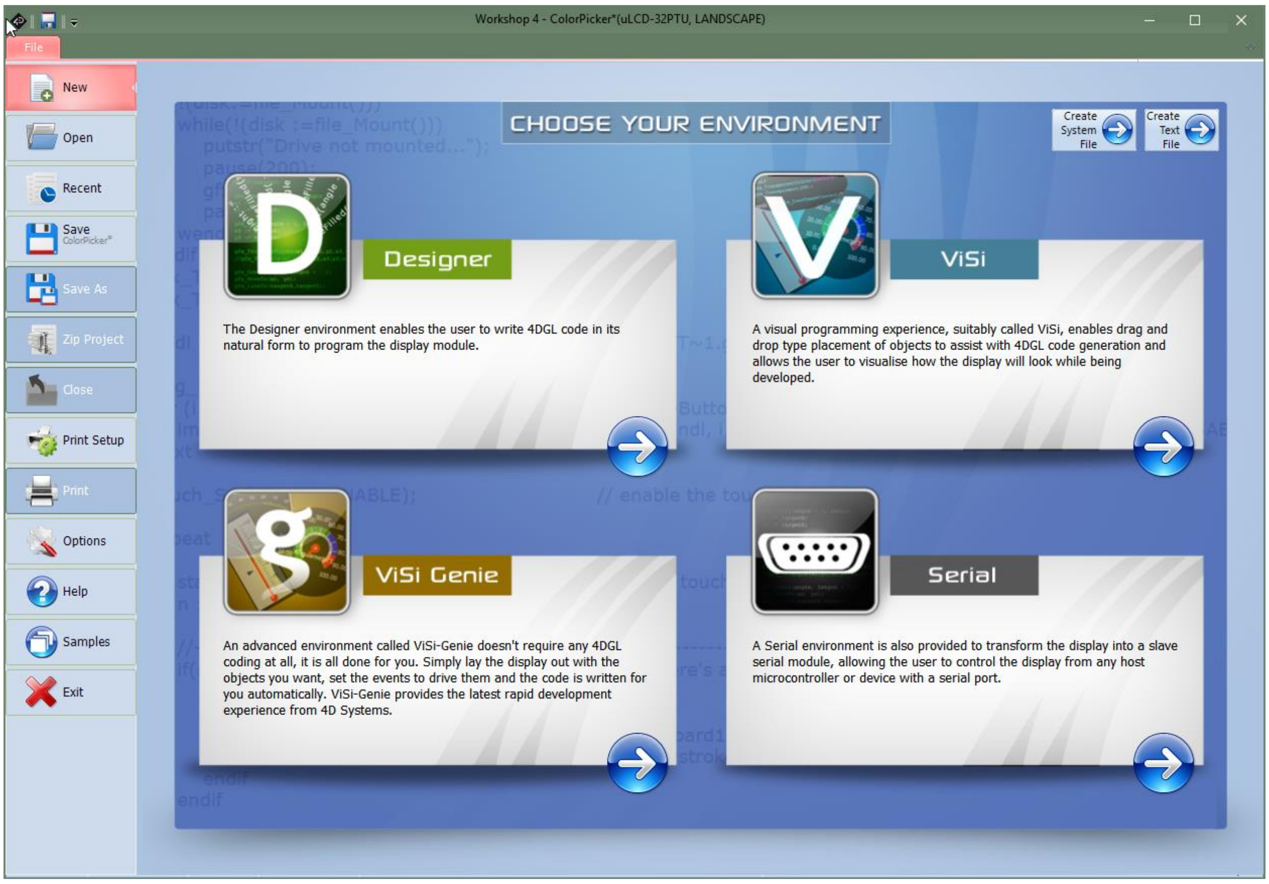
Gen4-uLCD-35DT 320x480
3.5" Gen4 Intelligent Display Module with embedded DIABLO16 processor w/ Resistive Touch panel

Gen4-uLCD-35DCT-CLB 320x480
3.5" Gen4 Intelligent Display Module with embedded DIABLO16 processor w/ Capacitive Touch panel & Cover Lens Bezel

Press **Next** to proceed:

4. Choose Your 4D Environment

The main window now asks for the environment for the project:



The Designer environment enables the user to write 4DGL code in its natural form to program the display module.

A visual programming experience, suitably called ViSi, enables drag and drop type placement of objects to assist with 4DGL code generation and allows the user to visualise how the display will look while being developed.

An advanced environment called ViSi-Genie doesn't require any 4DGL coding at all, it is all done for you. Simply lay the display out with the objects you want, set the events to drive them and the code is written for you automatically. ViSi-Genie provides the latest rapid development experience from 4D Systems.

A Serial environment is also provided to transform the display into a slave serial module, allowing the user to control the display from any host microcontroller or device with a serial port.

Four main environment options are available:

- Designer,
- ViSi,
- ViSi-Genie,
- Serial

…and two editor options:

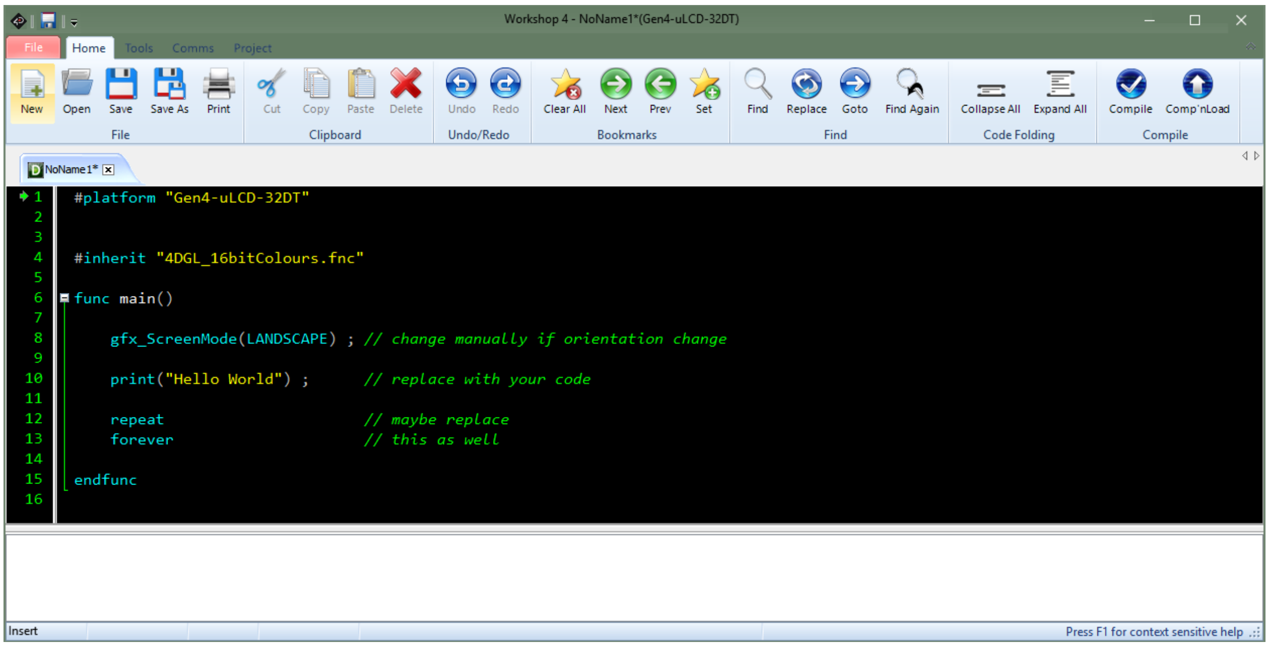
- Create System File,
- Create Text File.

Each option opens a customised environment with specific commands and controls. Different projects using different modules and environments can be opened simultaneously on Workshop4. The toolbar ribbon will adapt to the project tab that is selected, to suit its environment.

4.1. Designer

Designer
The Designer environment enables the user to write 4DGL code in its natural form to program the display module.

The **Designer** environment enables the user to write 4DGL code in its natural form to program the display module. 4DGL is a graphics-oriented language allowing rapid application development, and the syntax structure was designed using elements of popular languages such as C, Basic, Pascal and others.



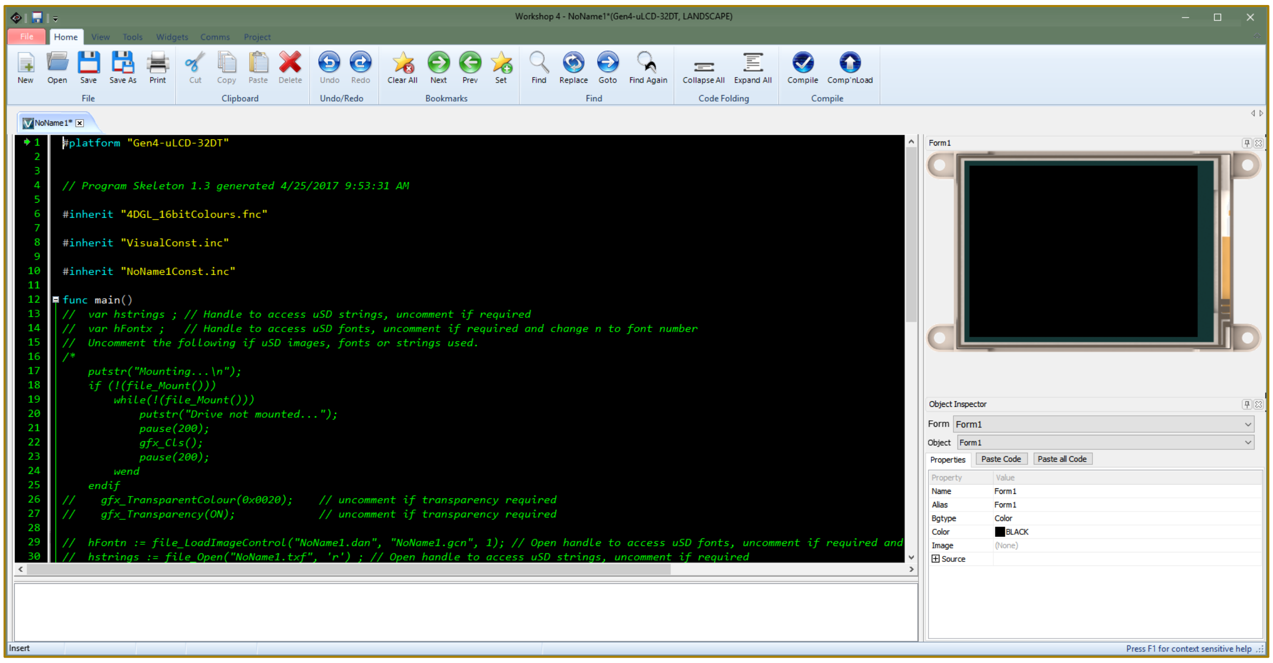
```
#platform "Gen4-uLCD-32DT"


#inherit "4DGL_16bitColours.fnc"

func main()

    gfx_ScreenMode(LANDSCAPE) ; // change manually if orientation change

    print("Hello World") ;      // replace with your code

    repeat                      // maybe replace
    forever                     // this as well

endfunc
```

To learn more about Designer, please to refer to the 4DGL Internal Functions Reference Manual relating to the processor used inside your display module, and 4DGL Programmers Reference Manual on the Workshop4 product page on the website, and the related application notes on the Application Note page.

4.2. ViSi

A visual programming experience, suitably called ViSi, enables drag and drop type placement of objects to assist with 4DGL code generation and allows the user to visualise how the display will look while being developed.

A visual programming experience, suitably called **ViSi**, enables drag-and-drop type placement of objects to assist with 4DGL code generation and allows the user to visualise how the display will look while being developed.



```
#platform "Gen4-uLCD-32DT"


// Program Skeleton 1.3 generated 4/25/2017 9:53:31 AM

#inherit "4DGL_16bitColours.fnc"

#inherit "VisualConst.inc"

#inherit "NoName1Const.inc"

func main()
//  var hstrings ; // Handle to access uSD strings, uncomment if required
//  var hFontx ;   // Handle to access uSD fonts, uncomment if required and change n to font number
//  Uncomment the following if uSD images, fonts or strings used.
/*
    putstr("Mounting...\n");
    if (!(file_Mount()))
        while(!(file_Mount()))
            putstr("Drive not mounted...");
            pause(200);
            gfx_Cls();
            pause(200);
        wend
    endif
//    gfx_TransparentColour(0x0020);    // uncomment if transparency required
//    gfx_Transparency(ON);             // uncomment if transparency required

//  hFontn := file_LoadImageControl("NoName1.dan", "NoName1.gcn", 1); // Open handle to access uSD fonts, uncomment if required and
//  hstrings := file_Open("NoName1.txf", 'r') ; // Open handle to access uSD strings, uncomment if required
```

Object Inspector

Form: Form1

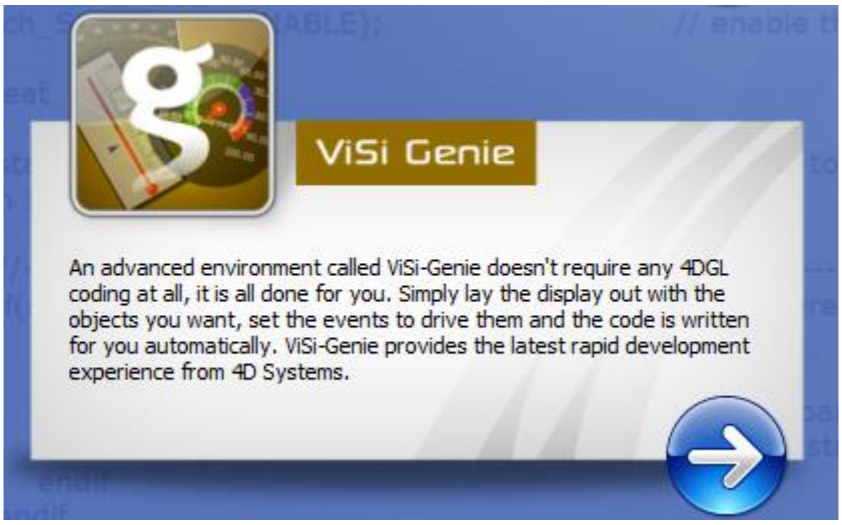
Object: Form1

Properties | Paste Code | Paste all Code

Property	Value
Name	Form1
Alias	Form1
Bgtype	Color
Color	■ BLACK
Image	(None)
⊞ Source	

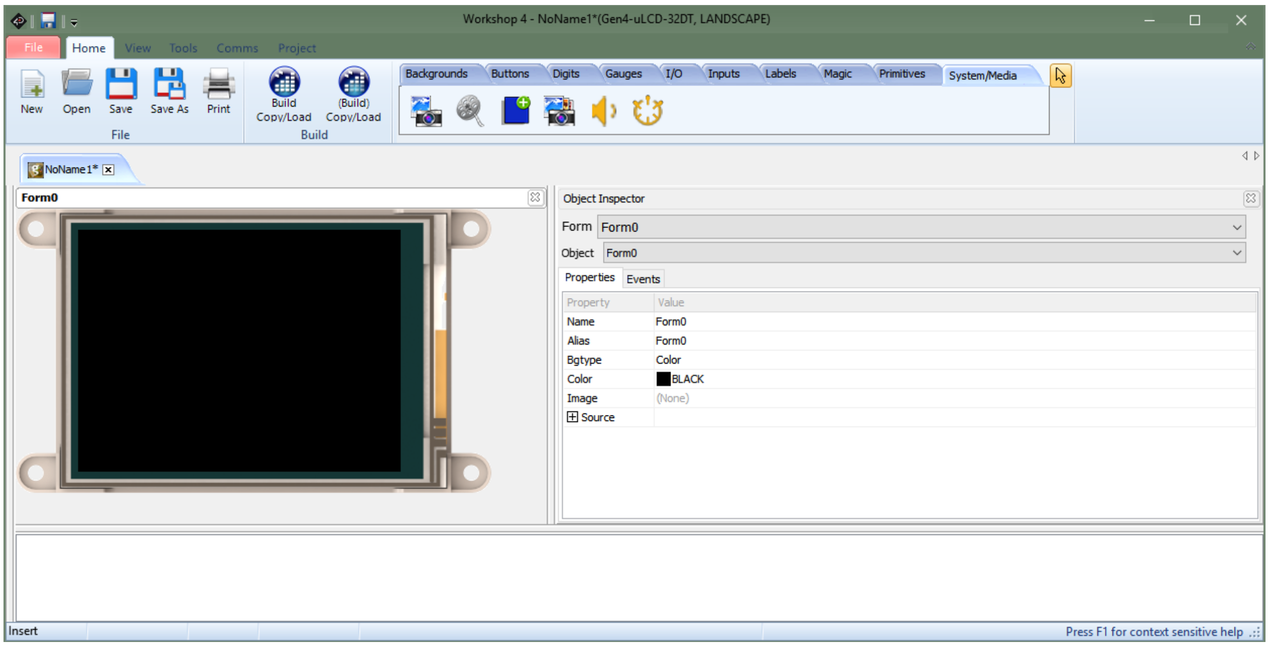
To learn more about ViSi, please to refer to the ViSi User Guide on the Workshop4 product page and the related application notes on the Application Note page on the website.

4.3. ViSi-Genie



An advanced environment called ViSi-Genie doesn't require any 4DGL coding at all, it is all done for you. Simply lay the display out with the objects you want, set the events to drive them and the code is written for you automatically. ViSi-Genie provides the latest rapid development experience from 4D Systems.

An advanced environment called **ViSi-Genie** doesn't require any 4DGL coding at all, it is all done for the user. By simply laying the display out with the wanted objects and setting the events to drive them, the code is written automatically. ViSi-Genie provides the latest rapid development experience from 4D Labs.



Form Form0

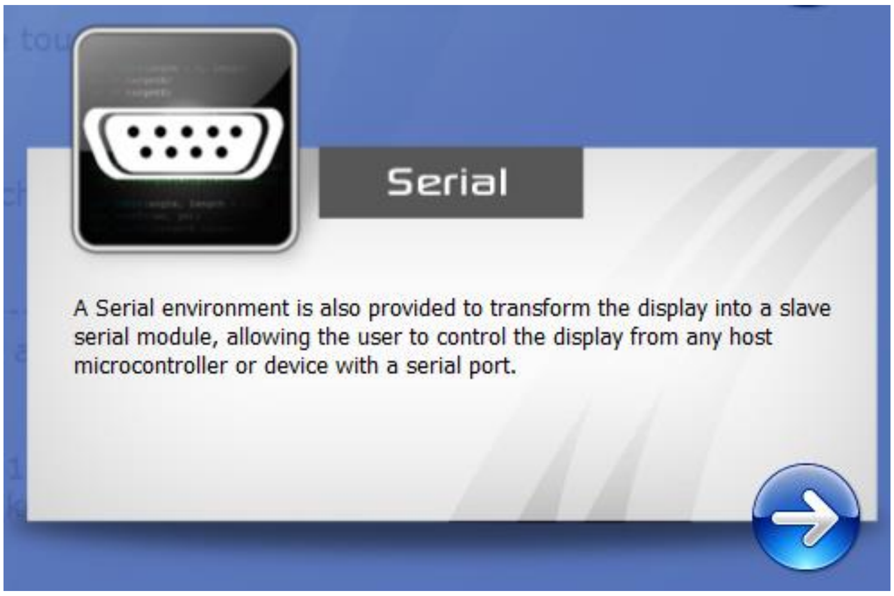
Object Form0

Property	Value
Name	Form0
Alias	Form0
Bgtype	Color
Color	■BLACK
Image	(None)
⊞ Source	

To learn more about ViSi-Genie, please to refer to ViSi-Genie User Guide on the Workshop4 product page and the related application notes on the Application Note page of the website.

Workshop4 PRO enables advanced features for **ViSi-Genie**, enabling 4DGL code to be used, allowing for increased customisation and flexibility on this already powerful environment. Please refer to the Application Notes for more information on Workshop4 PRO (specifically Genie Magic and Smart Widgets).

4.4. Serial



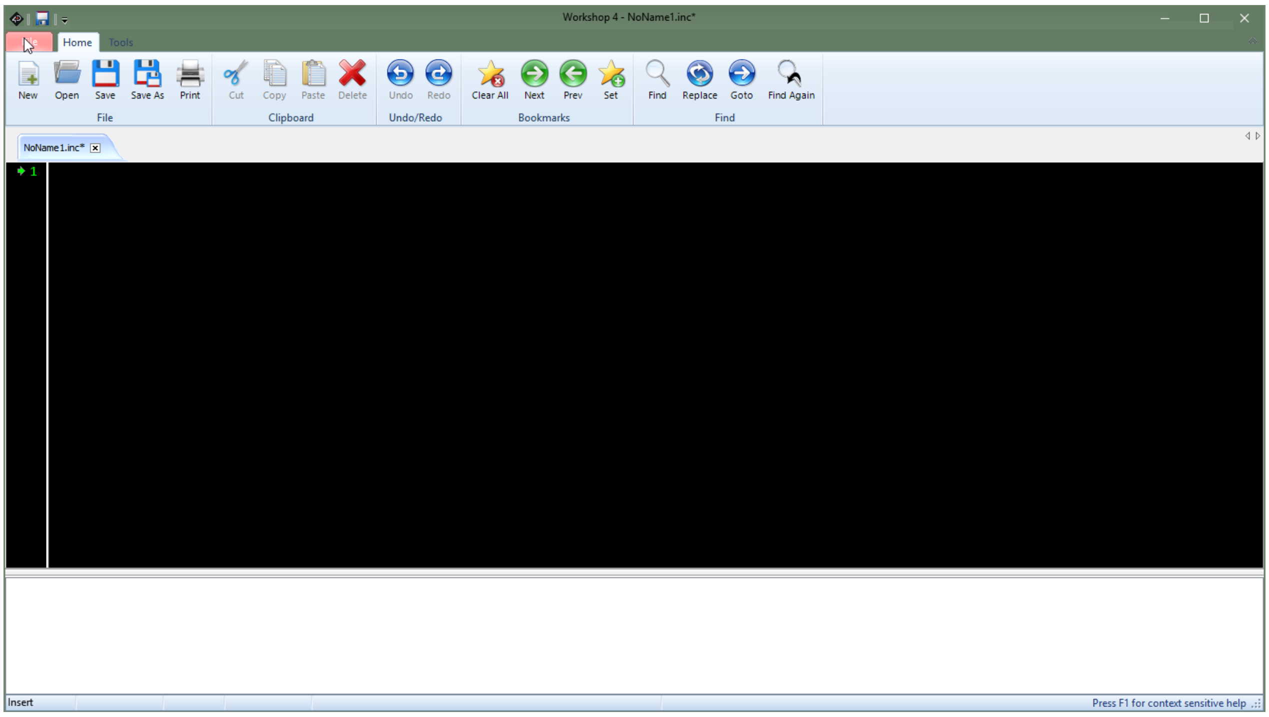
A Serial environment is also provided to transform the display into a slave serial module, allowing the user to control the display from any host microcontroller or device with a serial port.

The **Serial** environment is provided to transform the module into a slave serial module, allowing communication and control from virtually any host micro-controller or device with a serial port.

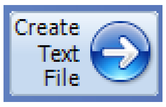
For detail on this environment and steps on how to make your display module run under the Serial environment, please refer to the Serial Command Set Reference Manual on the Workshop4 product page of the website, and the related application notes on the Application Note page.

4.5. Create System File

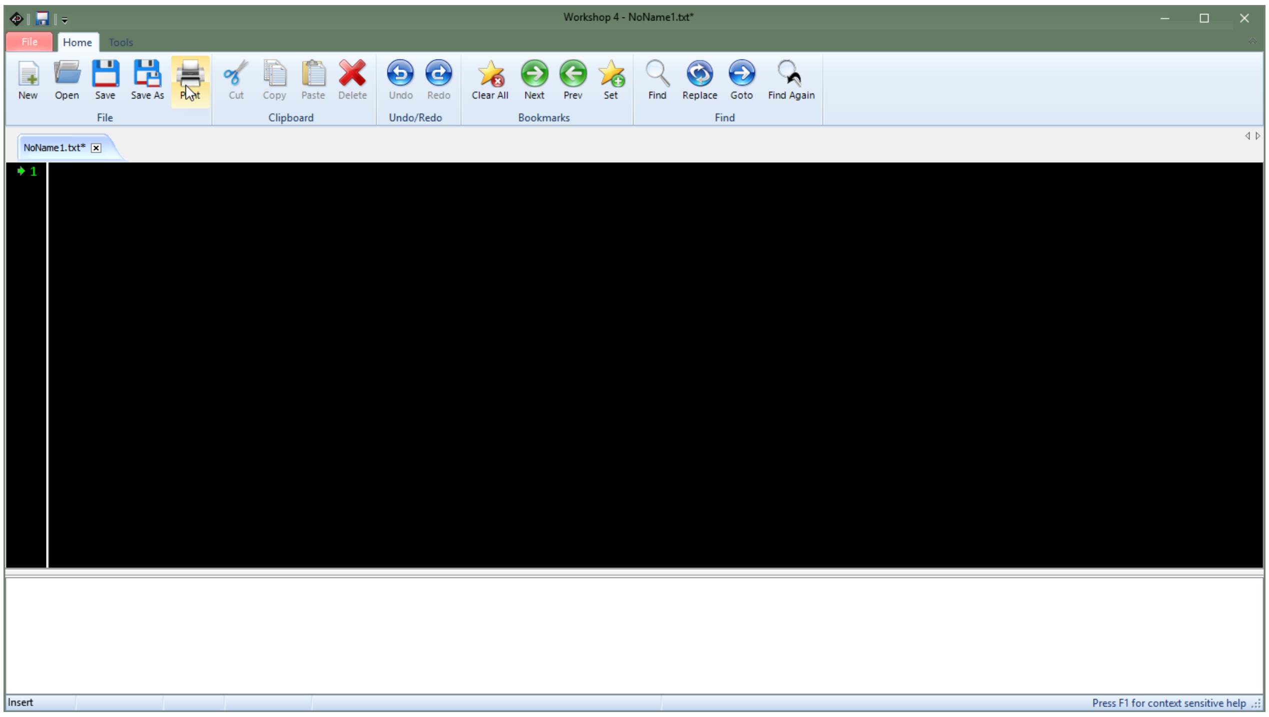
The **Create System File** option provides an editor for all 4DGL-related projects, so a user can create or edit a 4DGL Include file, 4DGL Library file, a Function or System file. These can then be included in the user's 4DGL code.



4.6. Create Text File

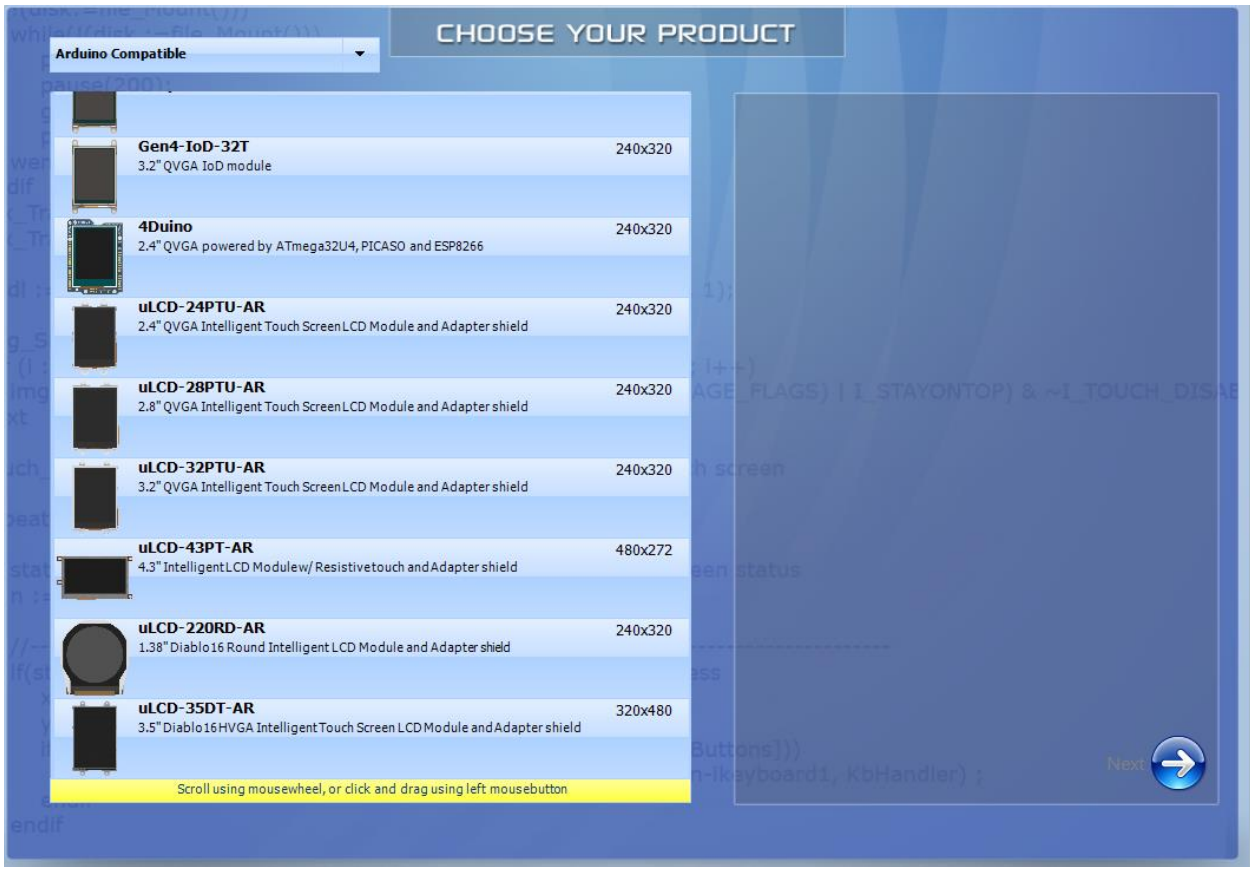


The **Create Text File** environment features a plain editor, suitable for writing basic documentation, application notes, data files or anything else requiring plain text files.



5. Choose Your Arduino Compatible Environment

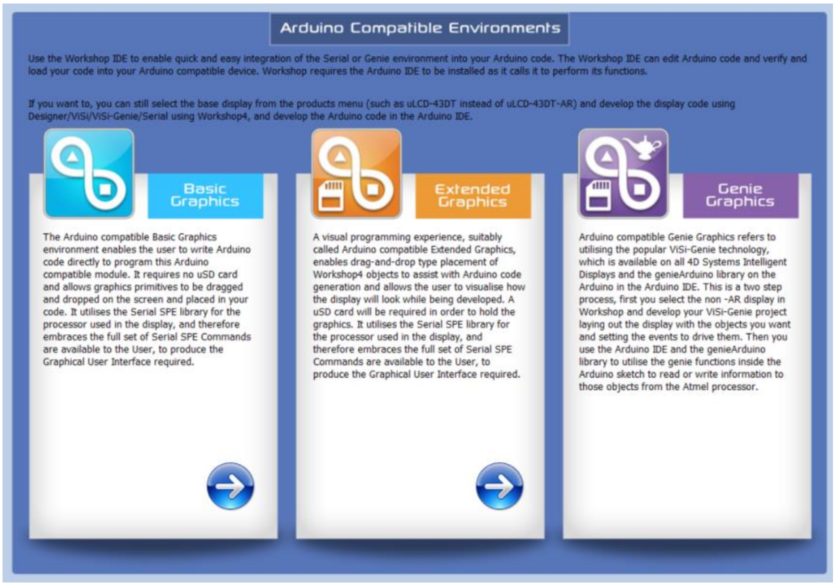
When choosing a 4D Systems product, you can see that there is a product group named **Arduino Compatible** which includes both the Internet of Displays (IoD) Modules and 4Duino. It also includes a wide range of 4D Systems' display modules setup with an Arduino board. If you prefer instead, you can use the standard 4D Environments along side the Arduino IDE.



Gen4-IoD-32T	3.2" QVGA IoD module	240x320
4Duino	2.4" QVGA powered by ATmega32U4, PICASO and ESP8266	240x320
uLCD-24PTU-AR	2.4" QVGA Intelligent Touch Screen LCD Module and Adapter shield	240x320
uLCD-28PTU-AR	2.8" QVGA Intelligent Touch Screen LCD Module and Adapter shield	240x320
uLCD-32PTU-AR	3.2" QVGA Intelligent Touch Screen LCD Module and Adapter shield	240x320
uLCD-43PT-AR	4.3" Intelligent LCD Module w/ Resistive touch and Adapter shield	480x272
uLCD-220RD-AR	1.38" Diablo16 Round Intelligent LCD Module and Adapter shield	240x320
uLCD-35DT-AR	3.5" Diablo16 HVGA Intelligent Touch Screen LCD Module and Adapter shield	320x480

Scroll using mousewheel, or click and drag using left mousebutton

Workshop4 allows the users to write their Arduino code and customize their 4D graphics user interface at the same time in a single integrated development environment.



Use the Workshop IDE to enable quick and easy integration of the Serial or Genie environment into your Arduino code. The Workshop IDE can edit Arduino code and verify and load your code into your Arduino compatible device. Workshop requires the Arduino IDE to be installed as it calls it to perform its functions.

If you want to, you can still select the base display from the products menu (such as uLCD-43DT instead of uLCD-43DT-AR) and develop the display code using Designer/ViSi/ViSi-Genie/Serial using Workshop4, and develop the Arduino code in the Arduino IDE.

The Arduino compatible Basic Graphics environment enables the user to write Arduino code directly to program this Arduino compatible module. It requires no uSD card and allows graphics primitives to be dragged and dropped on the screen and placed in your code. It utilises the Serial SPE library for the processor used in the display, and therefore embraces the full set of Serial SPE Commands are available to the User, to produce the Graphical User Interface required.

A visual programming experience, suitably called Arduino compatible Extended Graphics, enables drag-and-drop type placement of Workshop4 objects to assist with Arduino code generation and allows the user to visualise how the display will look while being developed. A uSD card will be required in order to hold the graphics. It utilises the Serial SPE library for the processor used in the display, and therefore embraces the full set of Serial SPE Commands are available to the User, to produce the Graphical User Interface required.

Arduino compatible Genie Graphics refers to utilising the popular ViSi-Genie technology, which is available on all 4D Systems Intelligent Displays and the genieArduino library on the Arduino in the Arduino IDE. This is a two step process, first you select the non -AR display in Workshop and develop your ViSi-Genie project laying out the display with the objects you want and setting the events to drive them. Then you use the Arduino IDE and the genieArduino library to utilise the genie functions inside the Arduino sketch to read or write information to those objects from the Atmel processor.

5.1. Basic Graphics

The Arduino compatible Basic Graphics environment enables the user to write Arduino code directly to program this Arduino compatible module. It requires no uSD card and allows graphics primitives to be dragged and dropped on the screen and placed in your code. It utilizes the Serial SPE library for the processor used in the display, and therefore embraces the full set of Serial SPE Commands are available to the User, to produce the Graphical User Interface required.

5.2. Extended Graphics

A visual programming experience, suitably called Arduino compatible Extended Graphics, enables drag-and-drop type placement of Workshop4 objects to assist with Arduino code generation and allows the user to visualize how the display will look while being developed. A uSD card will be required in order to hold the graphics. It utilizes the Serial SPE library for the processor used in the display, and therefore embraces the full set of Serial SPE Commands are available to the User, to produce the Graphical User Interface required.

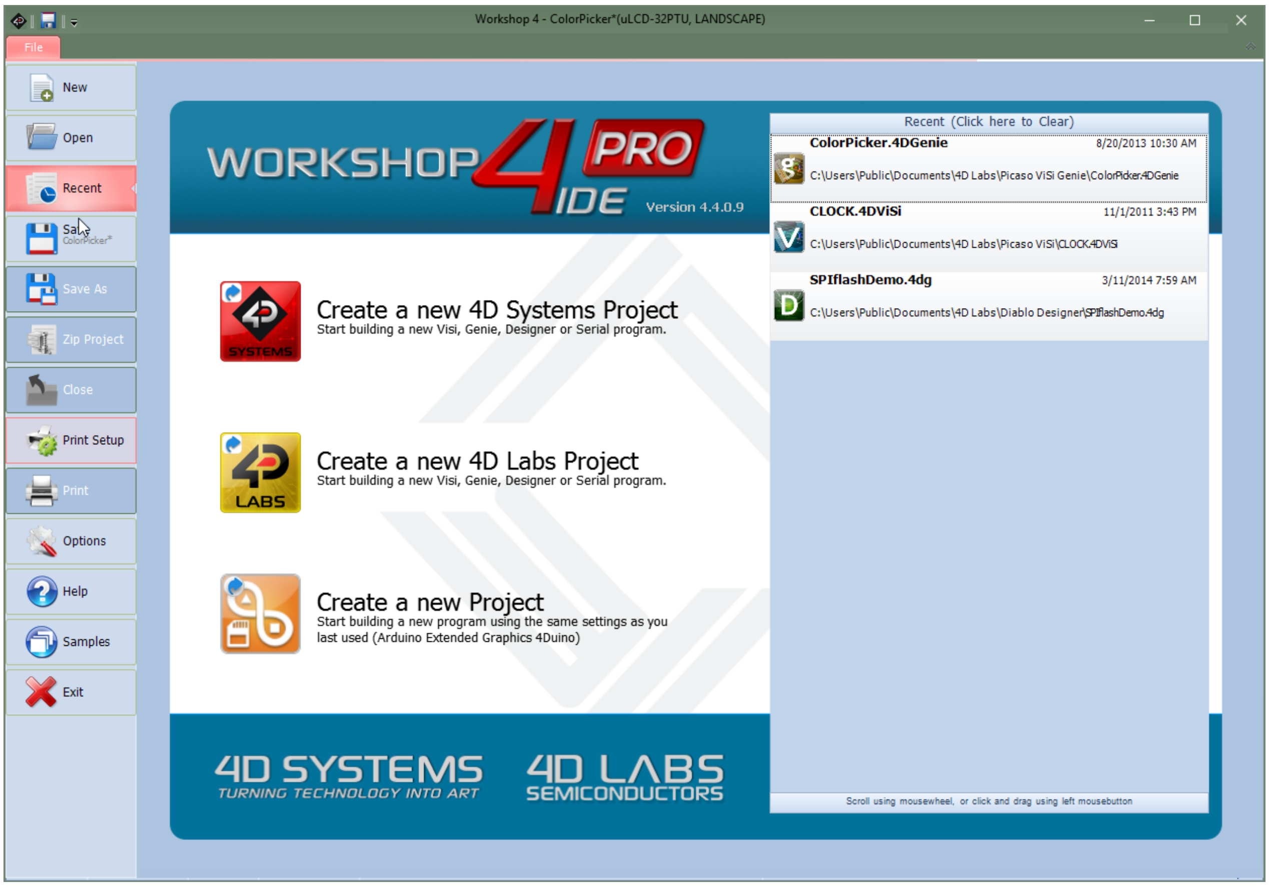
5.3. Genie Graphics

This is not an environment as such, but the description explains how to achieve Genie with an Arduino compatible 4D Module, or Kit.

Simply select the base module without the -AR extension from Workshop4 and utilize the standard ViSi-Genie environment with our genieArduino library and use the Arduino IDE for the Arduino development.

6. Common File Menu

The **File** Menu is the first menu and common to all environments…



Recent (Click here to Clear)

ColorPicker.4DGenie 8/20/2013 10:30 AM
C:\Users\Public\Documents\4D Labs\Picaso ViSi Genie\ColorPicker.4DGenie

CLOCK.4DViSi 11/1/2011 3:43 PM
C:\Users\Public\Documents\4D Labs\Picaso ViSi\CLOCK.4DViSi

SPIflashDemo.4dg 3/11/2014 7:59 AM
C:\Users\Public\Documents\4D Labs\Diablo Designer\SPIflashDemo.4dg

Create a new 4D Systems Project
Start building a new Visi, Genie, Designer or Serial program.

Create a new 4D Labs Project
Start building a new Visi, Genie, Designer or Serial program.

Create a new Project
Start building a new program using the same settings as you last used (Arduino Extended Graphics 4Duino)

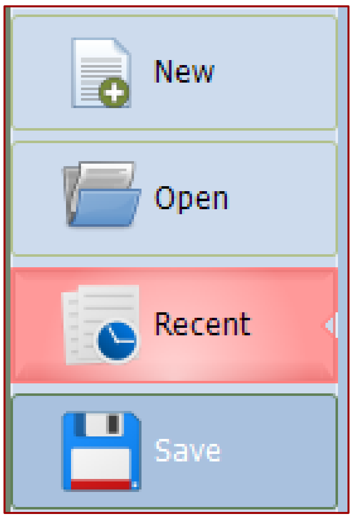
Scroll using mousewheel, or click and drag using left mousebutton

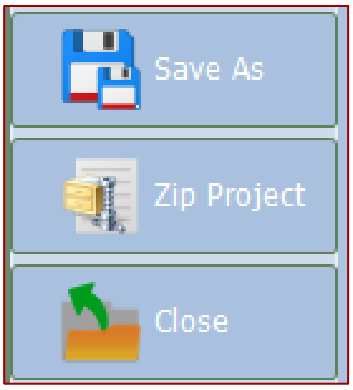
…with various buttons relating to the project that is open (or greyed out if no project is open):

- File-related buttons,
- Print-related buttons,
- And miscellaneous buttons, such as Help, Options and Samples.

6.1. File-Related Buttons

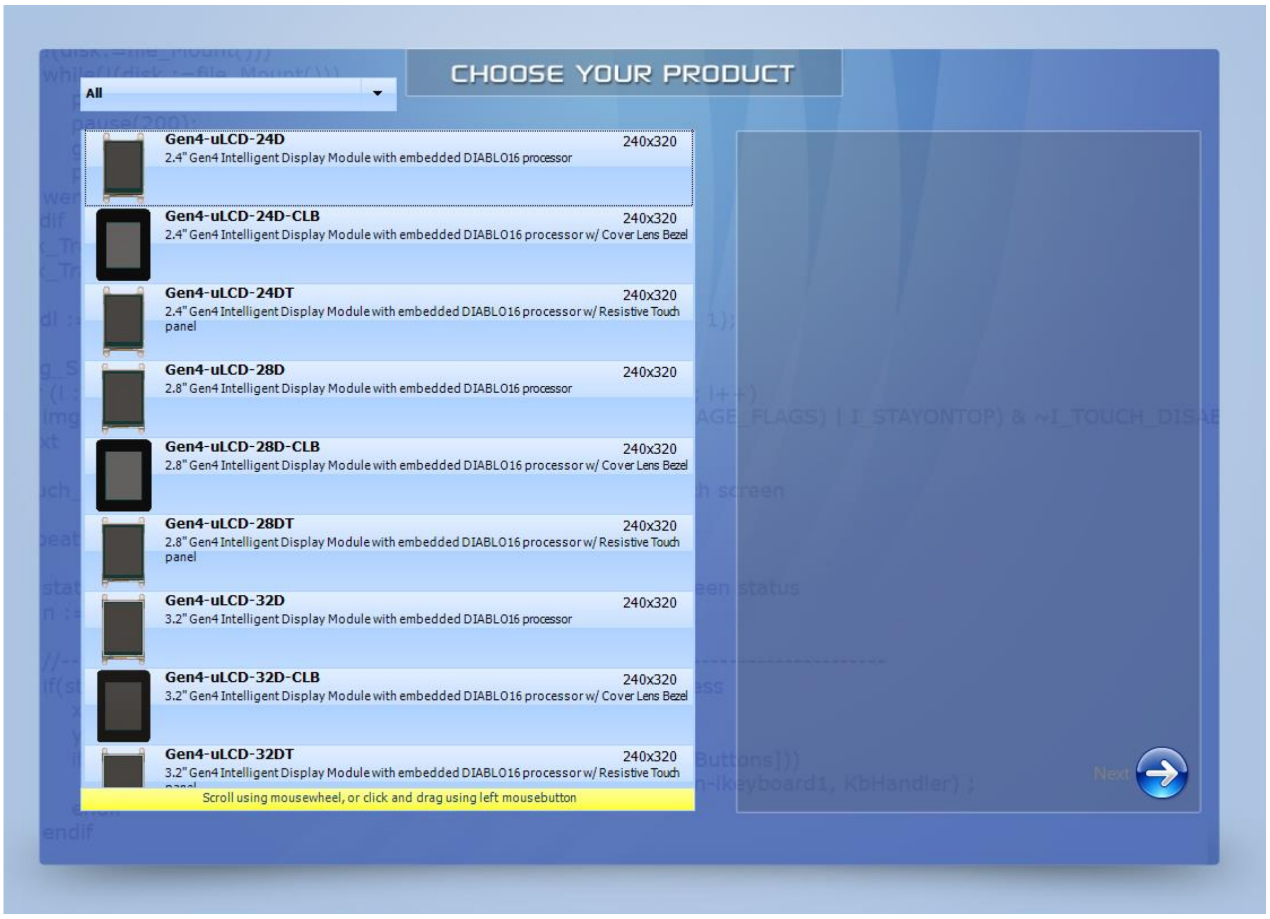
The buttons include all the actions related to projects: New, Open, Recent, Save, Save As, Zip Project and Close project.





The options include:

Click **New** to create a new project. A dialogue window asks for the screen and the kind of project.



Gen4-uLCD-24D 240x320
2.4" Gen4 Intelligent Display Module with embedded DIABLO16 processor

Gen4-uLCD-24D-CLB 240x320
2.4" Gen4 Intelligent Display Module with embedded DIABLO16 processor w/ Cover Lens Bezel

Gen4-uLCD-24DT 240x320
2.4" Gen4 Intelligent Display Module with embedded DIABLO16 processor w/ Resistive Touch panel

Gen4-uLCD-28D 240x320
2.8" Gen4 Intelligent Display Module with embedded DIABLO16 processor

Gen4-uLCD-28D-CLB 240x320
2.8" Gen4 Intelligent Display Module with embedded DIABLO16 processor w/ Cover Lens Bezel

Gen4-uLCD-28DT 240x320
2.8" Gen4 Intelligent Display Module with embedded DIABLO16 processor w/ Resistive Touch panel

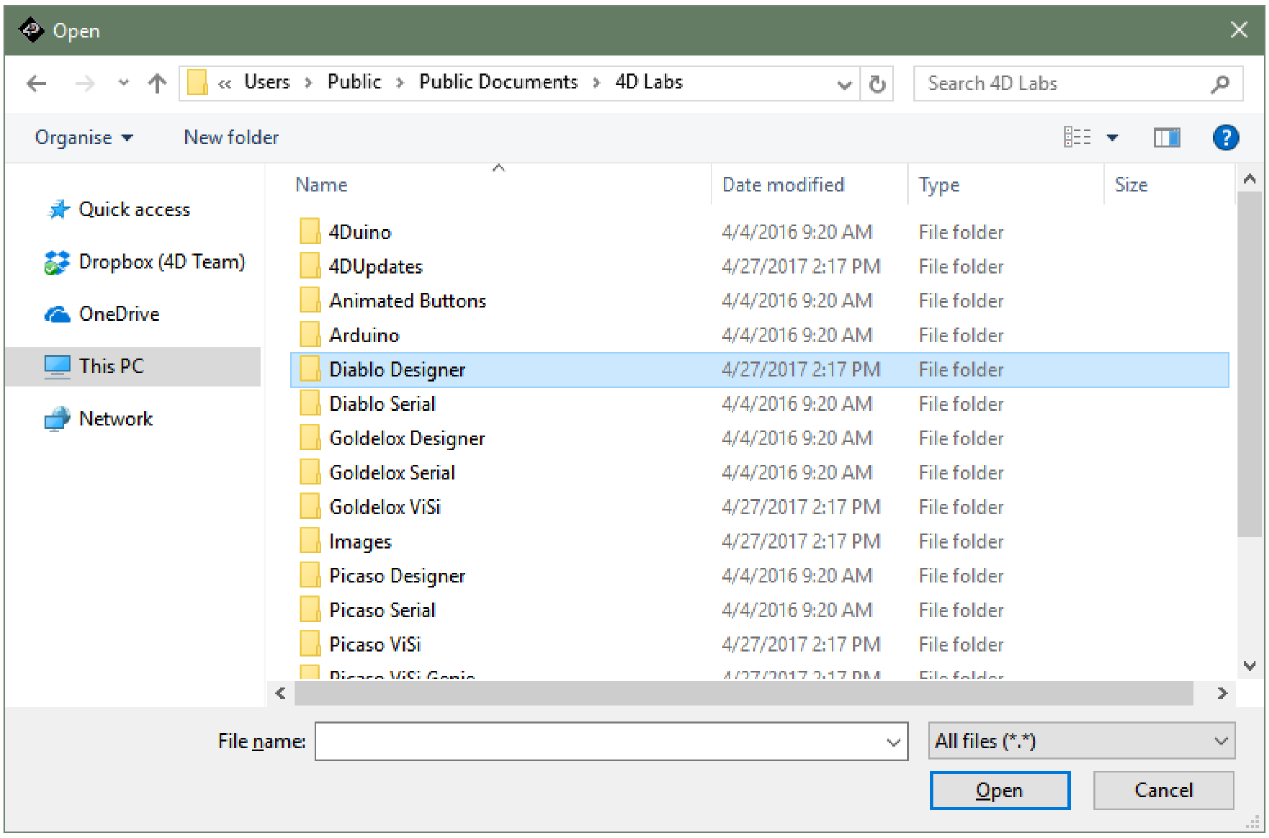
Gen4-uLCD-32D 240x320
3.2" Gen4 Intelligent Display Module with embedded DIABLO16 processor

Gen4-uLCD-32D-CLB 240x320
3.2" Gen4 Intelligent Display Module with embedded DIABLO16 processor w/ Cover Lens Bezel

Gen4-uLCD-32DT 240x320
3.2" Gen4 Intelligent Display Module with embedded DIABLO16 processor w/ Resistive Touch

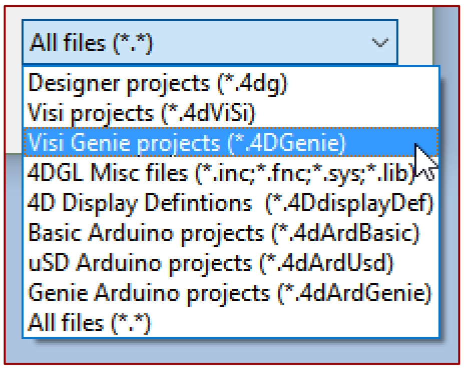
Scroll using mousewheel, or click and drag using left mousebutton

Click **Open** to browse and open an existing project. A standard Open window opens.

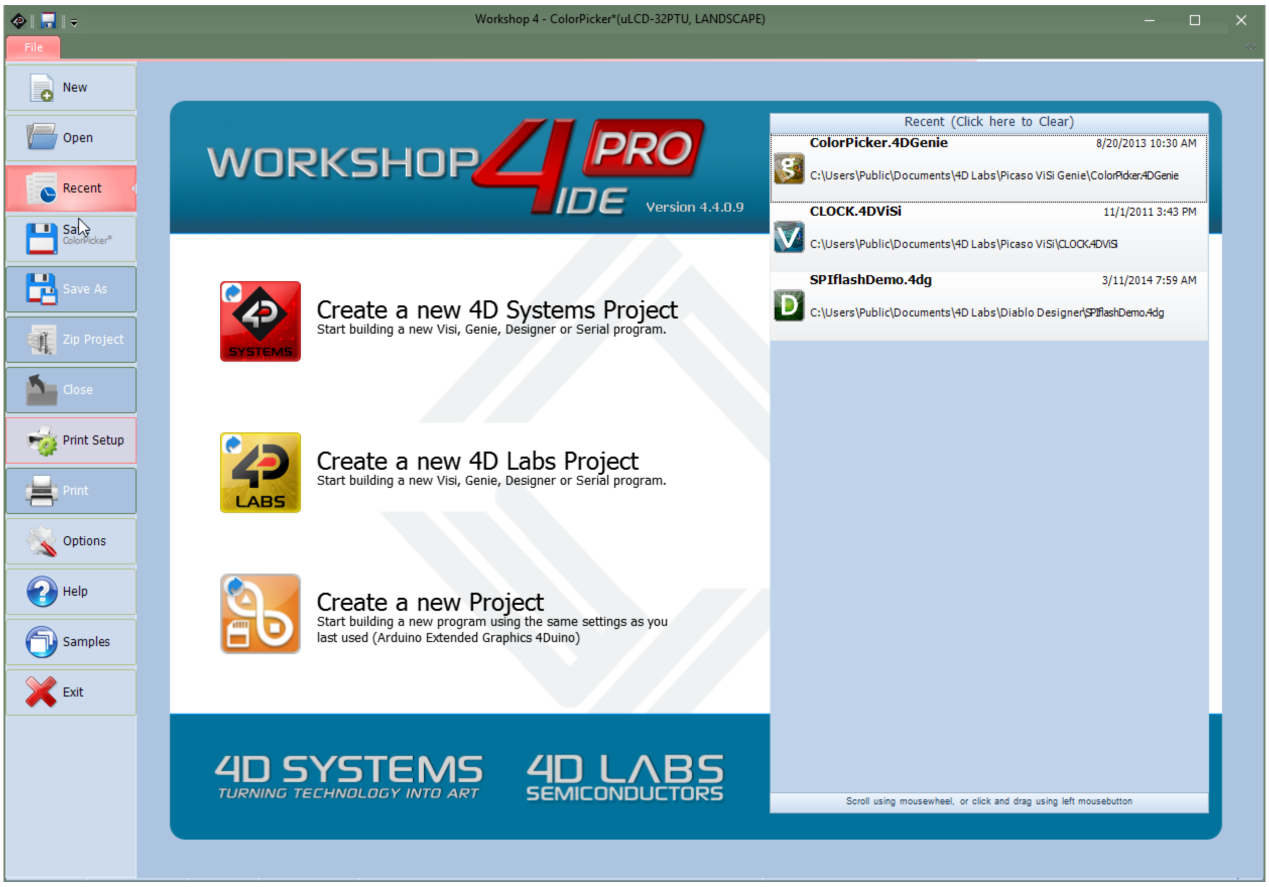


Name	Date modified	Type	Size
4Duino	4/4/2016 9:20 AM	File folder	
4DUpdates	4/27/2017 2:17 PM	File folder	
Animated Buttons	4/4/2016 9:20 AM	File folder	
Arduino	4/4/2016 9:20 AM	File folder	
Diablo Designer	4/27/2017 2:17 PM	File folder	
Diablo Serial	4/4/2016 9:20 AM	File folder	
Goldelox Designer	4/4/2016 9:20 AM	File folder	
Goldelox Serial	4/4/2016 9:20 AM	File folder	
Goldelox ViSi	4/27/2017 2:17 PM	File folder	
Images	4/27/2017 2:17 PM	File folder	
Picaso Designer	4/4/2016 9:20 AM	File folder	
Picaso Serial	4/4/2016 9:20 AM	File folder	
Picaso ViSi	4/27/2017 2:17 PM	File folder	

Select **ViSi-Genie projects** among the different kinds listed below to load a ViSi-Genie project:

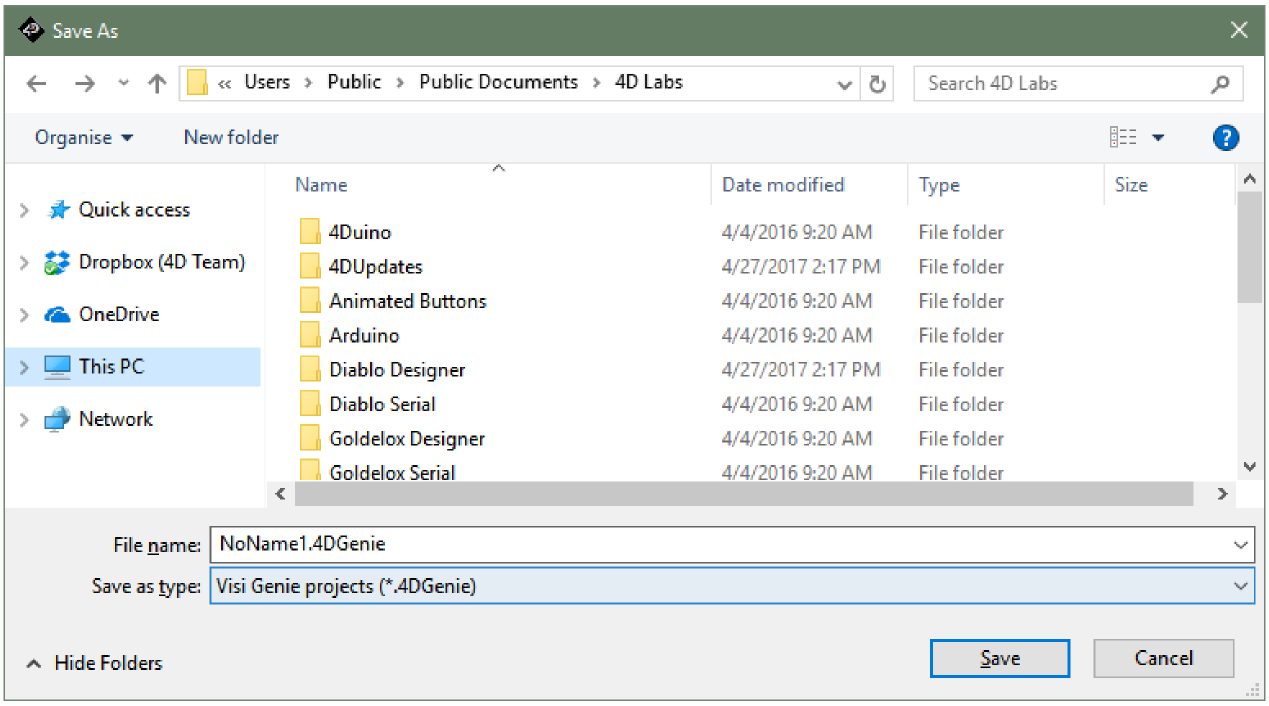

All files (*.*)
Designer projects (*.4dg)
Visi projects (*.4dViSi)
Visi Genie projects (*.4DGenie)
4DGL Misc files (*.inc;*.fnc;*.sys;*.lib)
4D Display Defintions (*.4DdisplayDef)
Basic Arduino projects (*.4dArdBasic)
uSD Arduino projects (*.4dArdUsd)
Genie Arduino projects (*.4dArdGenie)
All files (*.*)

Click **Recent** to list the recently accessed files and click on the project to open it.


Workshop 4 - ColorPicker*(uLCD-32PTU, LANDSCAPE)
File
New
Open
Recent
Save
ColorPicker*
Save As
Zip Project
Close
Print Setup
Print
Options
Help
Samples
Exit
WORKSHOP 4 PRO IDE
Version 4.4.0.9
Create a new 4D Systems Project
Start building a new Visi, Genie, Designer or Serial program.
Create a new 4D Labs Project
Start building a new Visi, Genie, Designer or Serial program.
Create a new Project
Start building a new program using the same settings as you last used (Arduino Extended Graphics 4Duino)
4D SYSTEMS
TURNING TECHNOLOGY INTO ART
4D LABS
SEMICONDUCTORS
Recent (Click here to Clear)
Scroll using mousewheel, or click and drag using left mousebutton

ColorPicker.4DGenie	8/20/2013 10:30 AM
C:\Users\Public\Documents\4D Labs\Picaso ViSi Genie\ColorPicker.4DGenie	
CLOCK.4DViSi	11/1/2011 3:43 PM
C:\Users\Public\Documents\4D Labs\Picaso ViSi\CLOCK.4DViSi	
SPIflashDemo.4dg	3/11/2014 7:59 AM
C:\Users\Public\Documents\4D Labs\Diablo Designer\SPIflashDemo.4dg	

Click **Save** to save the modified projects.
Click **Save As** to create a copy of already saved project and give it a new name. A standard Save window opens and asks for the location and the name:

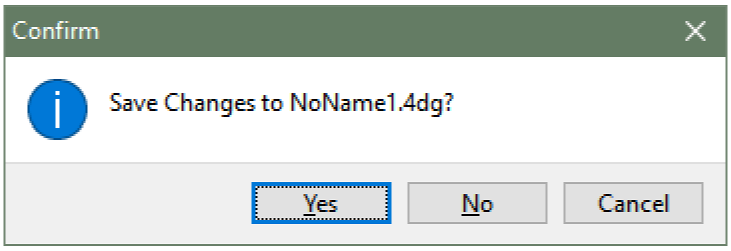


Name	Date modified	Type	Size
4Duino	4/4/2016 9:20 AM	File folder	
4DUpdates	4/27/2017 2:17 PM	File folder	
Animated Buttons	4/4/2016 9:20 AM	File folder	
Arduino	4/4/2016 9:20 AM	File folder	
Diablo Designer	4/27/2017 2:17 PM	File folder	
Diablo Serial	4/4/2016 9:20 AM	File folder	
Goldelox Designer	4/4/2016 9:20 AM	File folder	
Goldelox Serial	4/4/2016 9:20 AM	File folder	

File name: NoName1.4DGenie
Save as type: Visi Genie projects (*.4DGenie)

Click **Zip Project** to make a compressed file out of the project. This is especially useful when then project is large and contains pictures and videos.

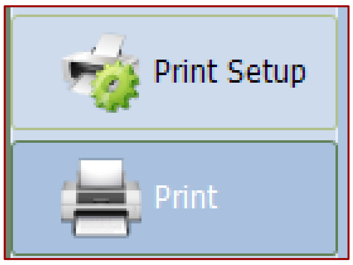
Click **Close** to close the current project. You will be prompted to save the changes to any modified project.


Confirm

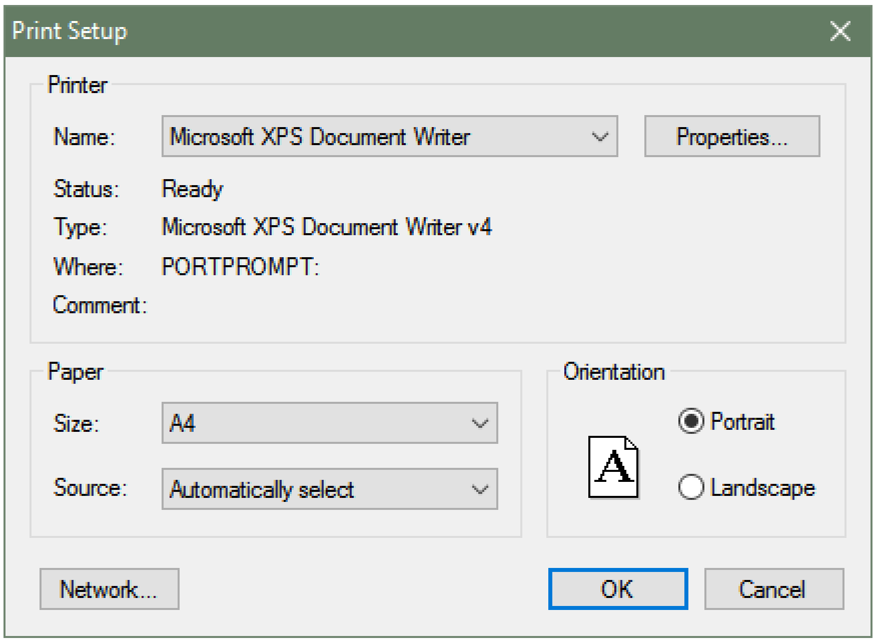
Save Changes to NoName1.4dg?

6.2. Print-Related Buttons

The buttons include all the actions related to print:



Click **Print Setup** to setup the printer:


Print Setup

Printer

Name: Microsoft XPS Document Writer

Status: Ready

Type: Microsoft XPS Document Writer v4

Where: PORTPROMPT:

Comment:

Paper

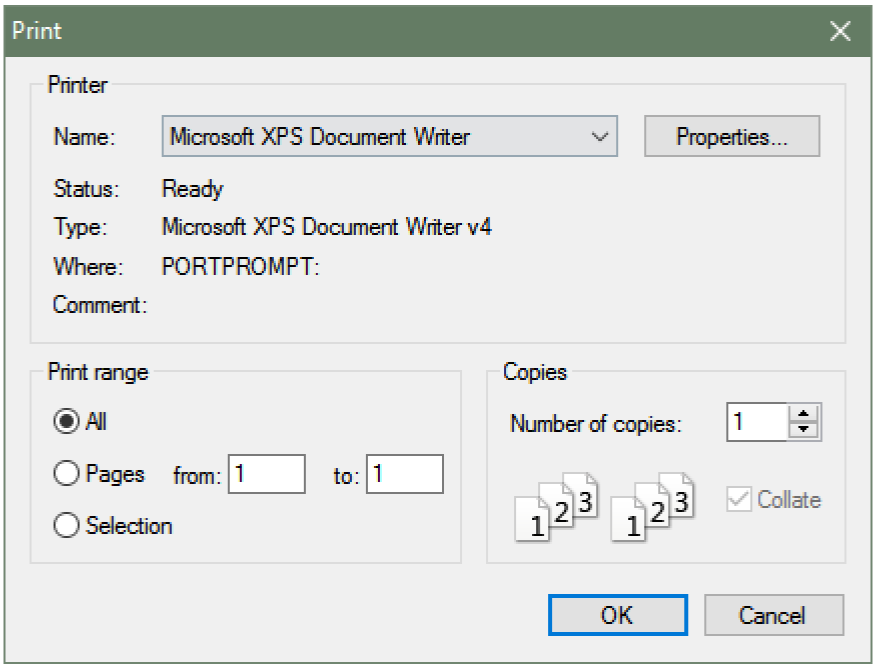
Size: A4

Source: Automatically select

Orientation

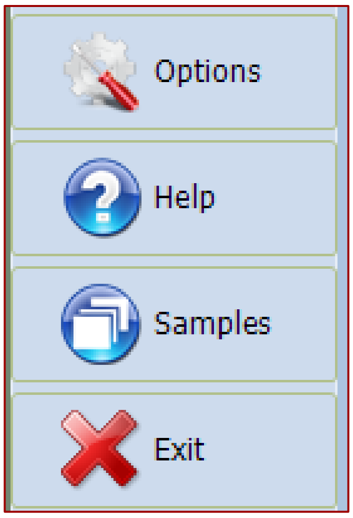
[✓ Portrait] Landscape

Click **Print** to print the project:

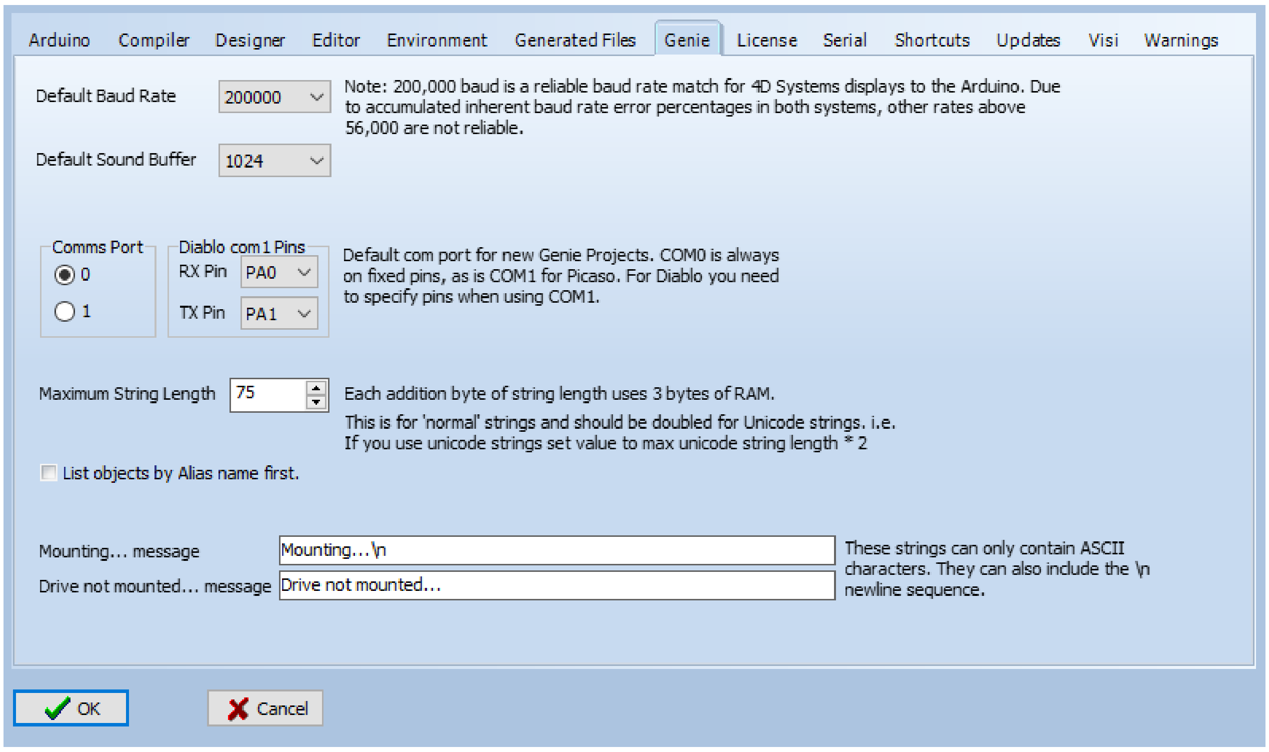


6.3. Miscellaneous Buttons

The Miscellaneous buttons include **Options**, **Help**, **Samples** and **Exit**.



Click **Options** to set the options for the default project. This is what a NEWLY created project will default to. The Baud Rate, Sound Buffer and Comms Port can be changed individually for a project in the Project Tab of the application itself. <u>The settings below are only to set the **defaults** for newly created projects</u>:



Default Baud Rate: 200000

Note: 200,000 baud is a reliable baud rate match for 4D Systems displays to the Arduino. Due to accumulated inherent baud rate error percentages in both systems, other rates above 56,000 are not reliable.

Default Sound Buffer: 1024

Comms Port: [✓ 0] 1

Diablo com1 Pins: RX Pin PA0, TX Pin PA1

Default com port for new Genie Projects. COM0 is always on fixed pins, as is COM1 for Picaso. For Diablo you need to specify pins when using COM1.

Maximum String Length: 75

Each addition byte of string length uses 3 bytes of RAM.
This is for 'normal' strings and should be doubled for Unicode strings. i.e.
If you use unicode strings set value to max unicode string length * 2

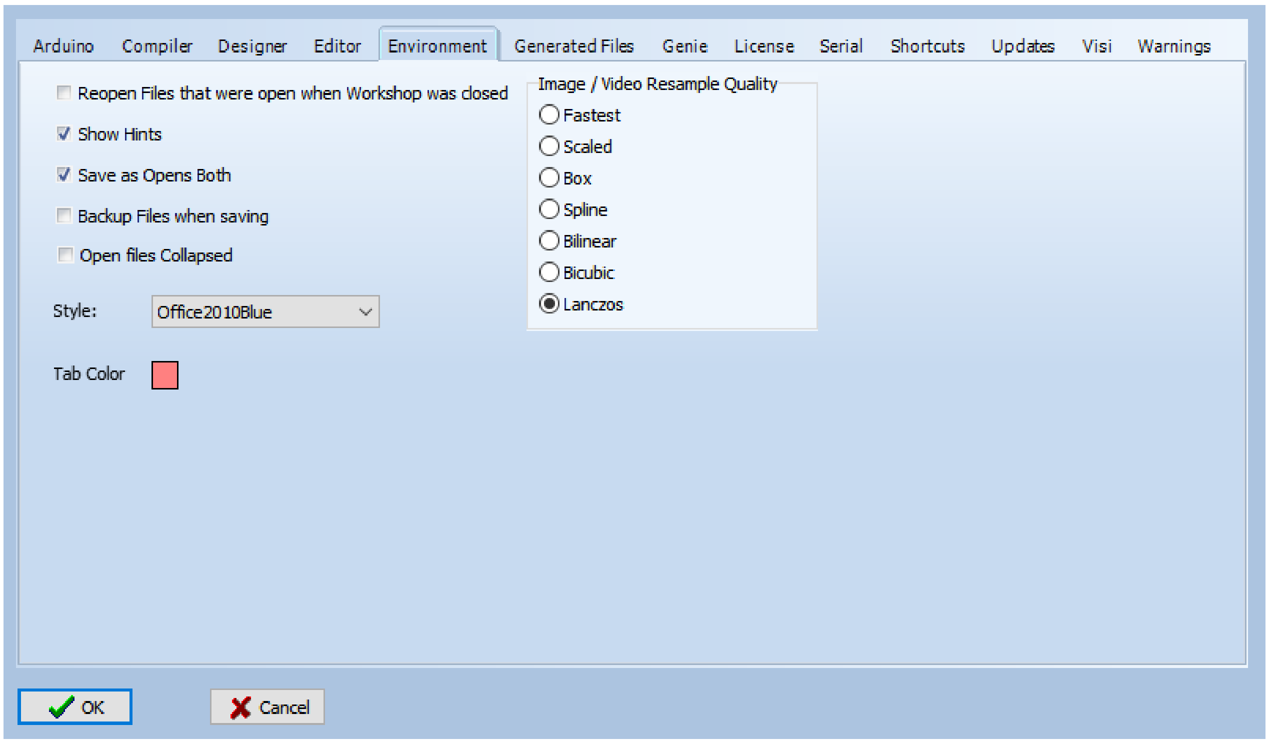
☐ List objects by Alias name first.

Mounting... message: Mounting...\n

Drive not mounted... message: Drive not mounted...

These strings can only contain ASCII characters. They can also include the \n newline sequence.

Options include different panes. Select "Environment".



☐ Reopen Files that were open when Workshop was closed
☑ Show Hints
☑ Save as Opens Both
☐ Backup Files when saving
☐ Open files Collapsed

Style: Office2010Blue

Tab Color: 🟥

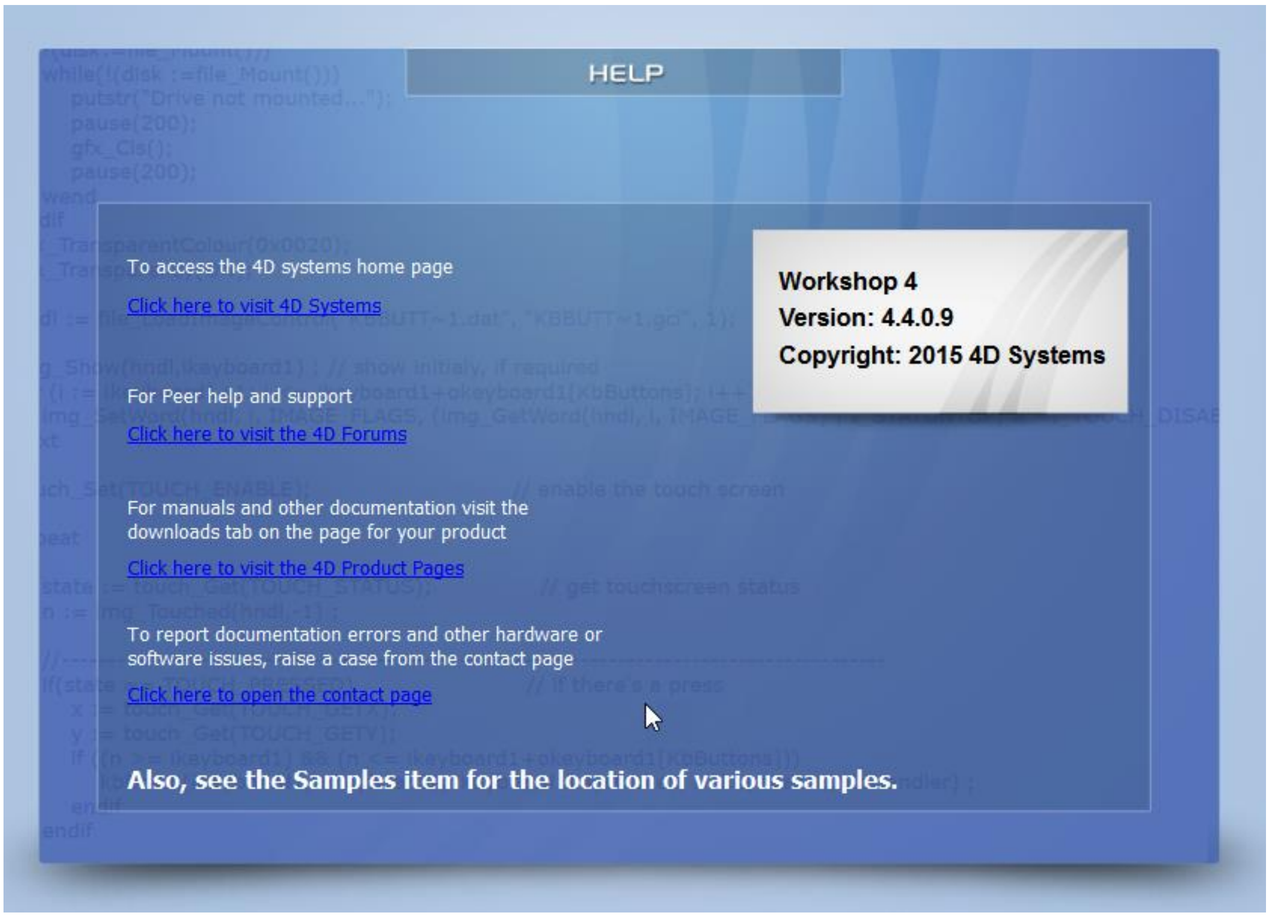
Image / Video Resample Quality: Fastest, Scaled, Box, Spline, Bilinear, Bicubic, [✓ Lanczos]

Select the **Style** among different possibilities.

The **Image / Video Resample Quality** selection brings different options:

- Fastest
- Scaled
- Box
- Spline
- Bilinear
- Bicubic
- Lanczos

Click **Help** to access help, with links to the 4D website:



HELP

To access the 4D systems home page

Click here to visit 4D Systems

For Peer help and support

Click here to visit the 4D Forums

For manuals and other documentation visit the downloads tab on the page for your product

Click here to visit the 4D Product Pages

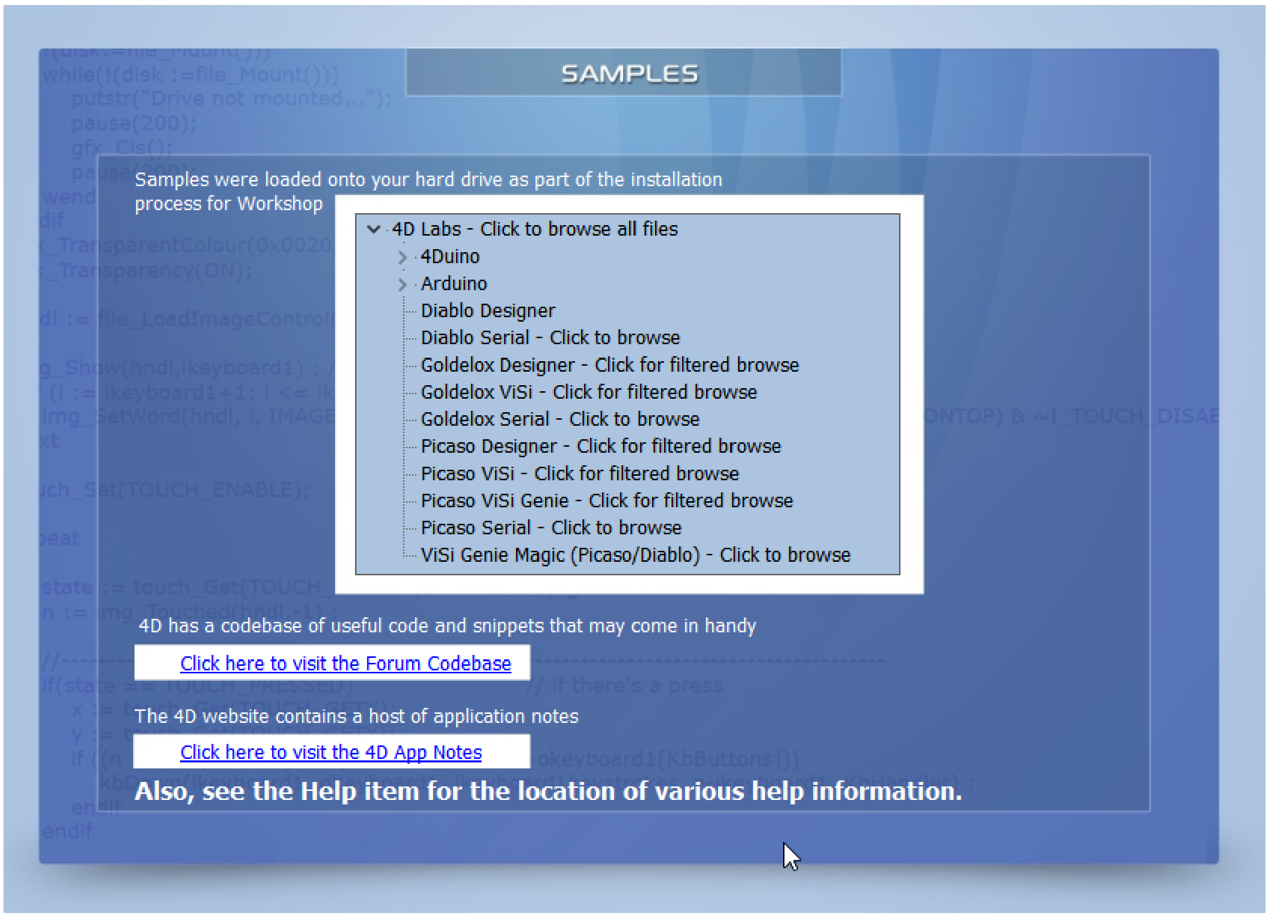
To report documentation errors and other hardware or software issues, raise a case from the contact page

Click here to open the contact page

Also, see the Samples item for the location of various samples.

Workshop 4
Version: 4.4.0.9
Copyright: 2015 4D Systems

Click **Samples** to access examples from 4D, with pre-defined filters:


SAMPLES

Samples were loaded onto your hard drive as part of the installation process for Workshop

- 4D Labs - Click to browse all files
  - 4Duino
  - Arduino
  - Diablo Designer
  - Diablo Serial - Click to browse
  - Goldelox Designer - Click for filtered browse
  - Goldelox ViSi - Click for filtered browse
  - Goldelox Serial - Click to browse
  - Picaso Designer - Click for filtered browse
  - Picaso ViSi - Click for filtered browse
  - Picaso ViSi Genie - Click for filtered browse
  - Picaso Serial - Click to browse
  - ViSi Genie Magic (Picaso/Diablo) - Click to browse

4D has a codebase of useful code and snippets that may come in handy

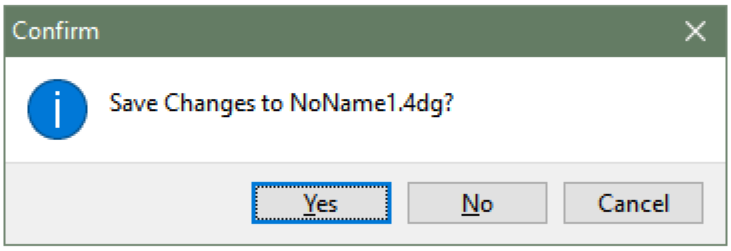
Click here to visit the Forum Codebase

The 4D website contains a host of application notes

Click here to visit the 4D App Notes

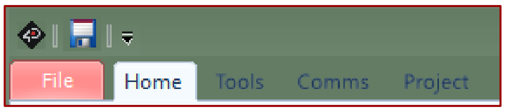
Also, see the Help item for the location of various help information.

Click **Exit** to close Workshop4. You will be prompted to save the changes to any modified project.



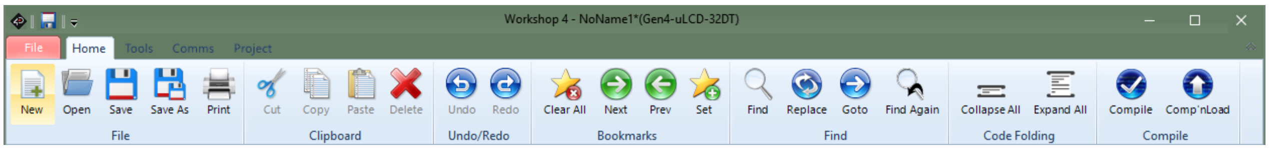
7. Designer Specific Menus

The Designer environment includes five menus:



7.1. Home Menu

The **Home** menu is the main menu.

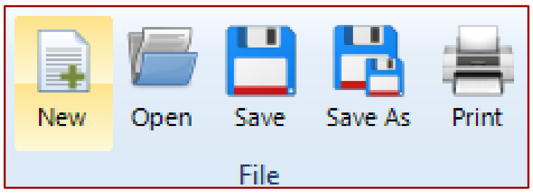


...with three groups of buttons:

- File-related commands,
- Build command,
- And the objects pane.

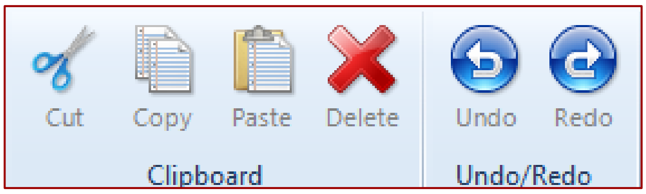
7.1.1. File-Related Buttons

The file-related buttons include the same commands as seen in the File menu: New, Open, Save, Save As and Print.



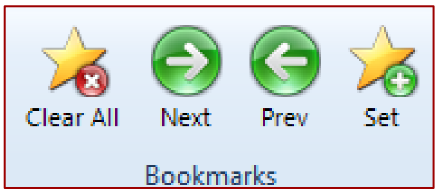
7.1.2. Code-Related Buttons

The code related buttons include the standard Windows commands of Cut, Copy, Paste, Delete, Undo and Redo.

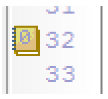


7.1.3. Bookmark Buttons

The bookmark buttons include **Set** a bookmark, go to **Next** or **Previous** bookmark and **Clear All** bookmarks.



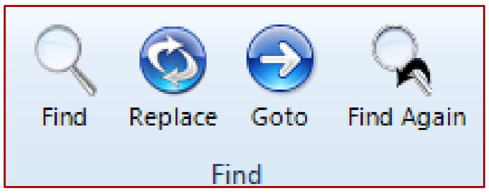
Bookmarks are shown close to the line number:



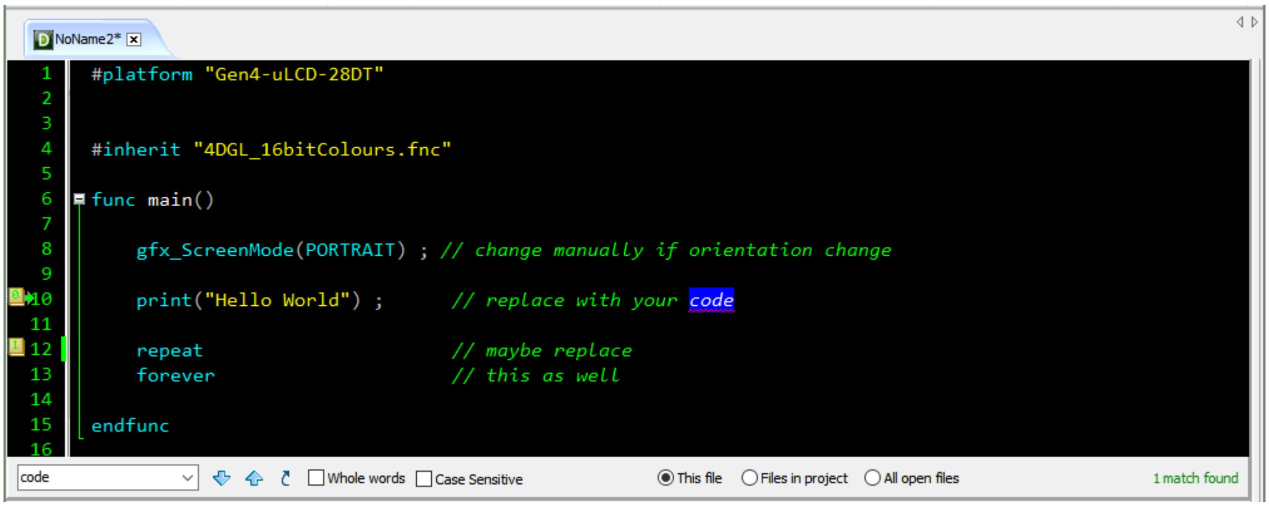
Bookmarks are especially useful for large projects.

7.1.4. Find and Replace Buttons

The find and replace buttons provide the basic features for code.



The **Find** button prompts for a string and highly it in the code:

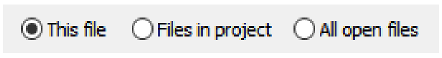


NoName2*

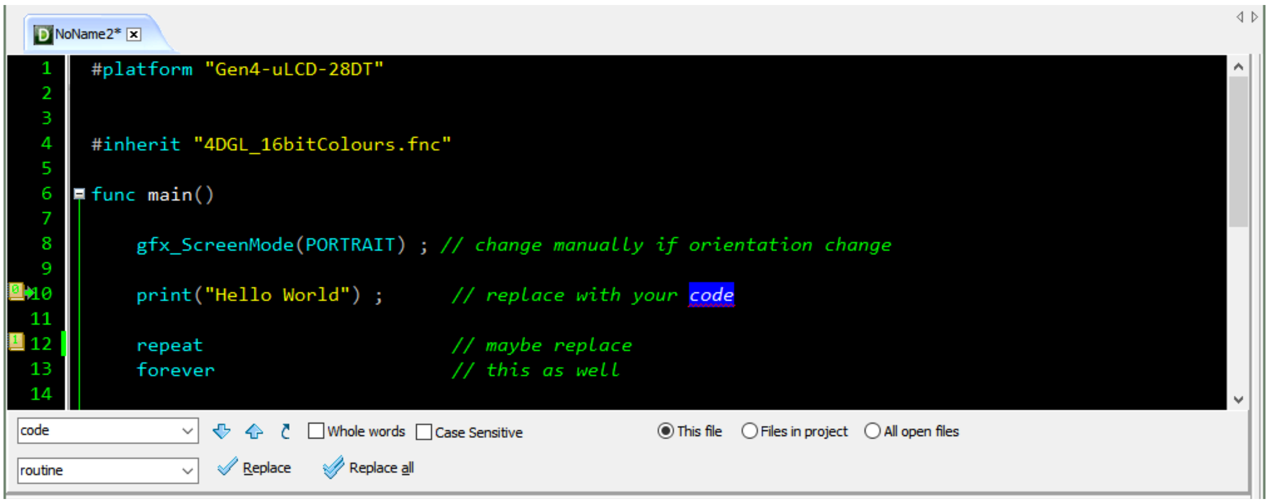
```
#platform "Gen4-uLCD-28DT"


#inherit "4DGL_16bitColours.fnc"

func main()

    gfx_ScreenMode(PORTRAIT) ; // change manually if orientation change

    print("Hello World") ;      // replace with your code

    repeat                      // maybe replace
    forever                     // this as well

endfunc
```

code [ ] Whole words [ ] Case Sensitive [✓ This file] Files in project All open files 1 match found

Use the up and down arrows to look for the previous and next occurrence. Check **Whole Words** and **Case Sensitive**. Choose between **This file** and **Files in progress** and **All Open Files**.



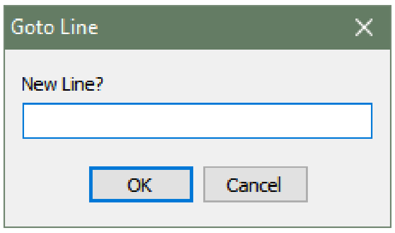
The **Replace** button searches for a string and exchanges it with another string:



```
#platform "Gen4-uLCD-28DT"


#inherit "4DGL_16bitColours.fnc"

func main()

    gfx_ScreenMode(PORTRAIT) ; // change manually if orientation change

    print("Hello World") ;      // replace with your code

    repeat                      // maybe replace
    forever                     // this as well
```

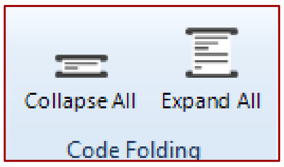
Same options as for Find apply.

The **Goto** button prompts for a line number:



7.1.5. Code Folding Buttons

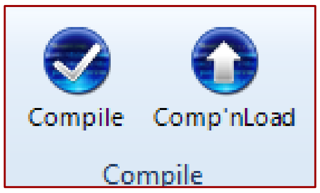
The code folding buttons allow to collapse or expand a function:



This is especially useful for large projects.

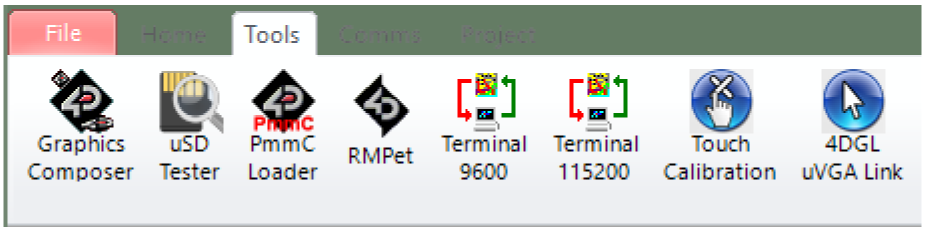
7.1.6. Compile Buttons

The **Compile** button launches the compilation of the project, can also be accessed with shortcut key Ctrl + F9.
The **Comp'nLoad** compiles and uploads the project to the screen, which has a shortcut key of F9.



Once a project has been loaded, if there are no changes, then the Comp'nLoad button will change to a Download button.

7.2. Tools Menu

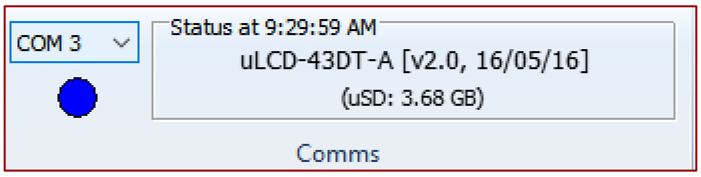


All the tools and utilities are grouped here:

- Click **Graphics Composer** to open the Graphics Composer tool. This tool is used for creating graphics for a Designer project. If graphics is needed for a Designer project, it is recommended to use ViSi instead.
- Clik **uSD Tester** to test the uSD card mounted on the display. Before clicking this button, make sure that the uSD card is mounted on the display module.
- Click **PmmC Loader** to start the PmmC Loader utility.
- Click **RMPet** to partition the uSD card. The uSD card must be mounted to the PC.
- 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.
- Click **Touch Calibration** to calibrate the touch on the display. This tool is not available for all display modules.
- Click **4DGL uVGA Link** to open a window which can interact with the uVGA, enabling a mouse and keyboard to be used with the uVGA(GFX) module. Demo code is available from Workshop4.

7.3. Comms Menu

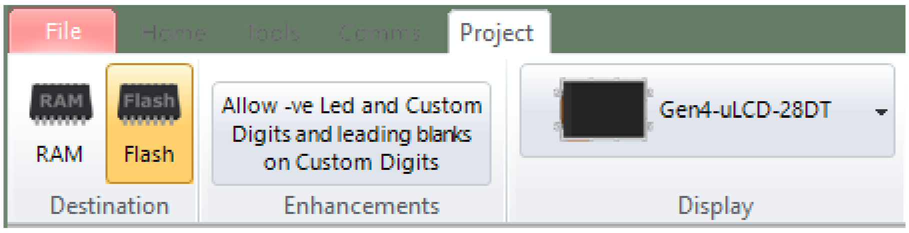
This menu is in charge of the communication port:



The use of this menu is described at the section Connect the Module.

7.4. Project Menu

The **Project** menu includes different parameters and options…

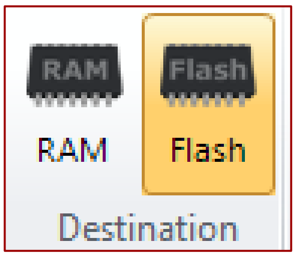


…with three groups of buttons:

- Destination,
- Enhancements,
- And Display selection.

7.4.1. Destination

The first group includes the options for the destination of the compiled code (not images/multimedia etc):

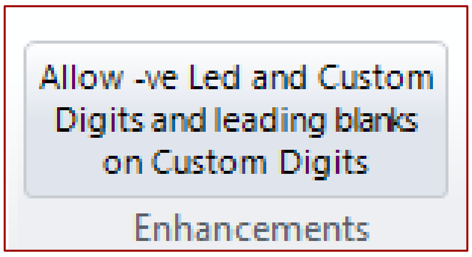


Select the **Destination** among two possibilities:

- **RAM** means the display must be connected during build and that the program will be downloaded to the display's RAM memory once compiled. If **RAM** is chosen as the destination, the program is lost when the display is turned off. This really is for testing while developing, to prevent the Flash cycles being used.
- **Flash** means the display must be connected during build and that the program will be downloaded to the display's flash memory once compiled. If **flash** is chosen as the destination, the program is retained and will be available after power cycling. This should be the default for normal use.

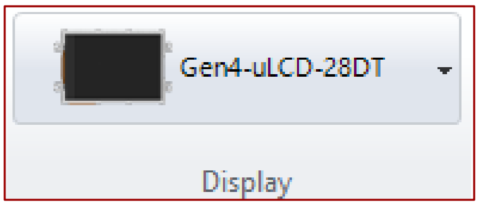
7.4.2. Enhancements

The second section contains a button for enabling the use of negative values for LED digits and custom digits objects and for enabling the use of leading blanks on custom digit objects.

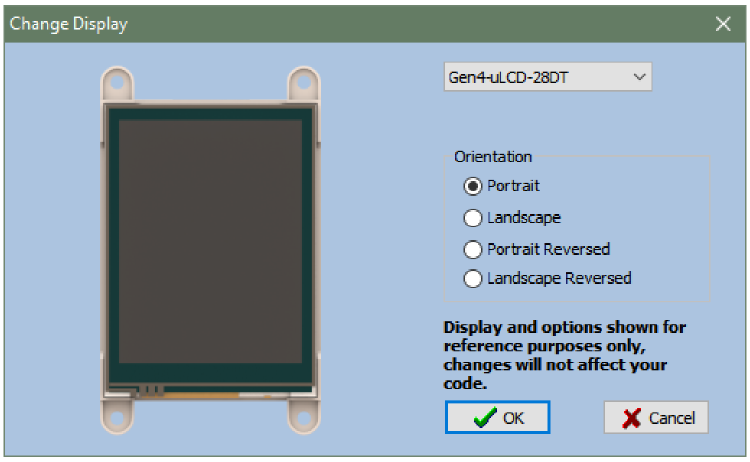

Allow -ve Led and Custom Digits and leading blanks on Custom Digits
Enhancements

7.4.3. Display Selection

The third section allows to select the screen. Clicking on the button…

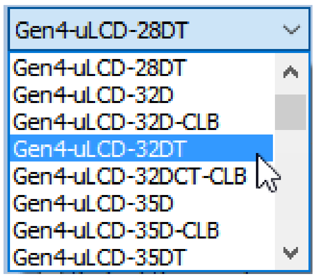


…opens a new window to select the screen, and adjust the orientation of the module to suit how the module will be mounted:

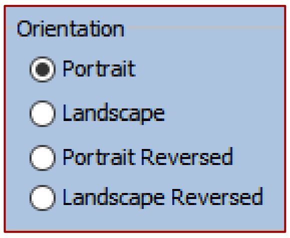


Display and options shown for reference purposes only, changes will not affect your code.

Select the screen from the drop-down list:



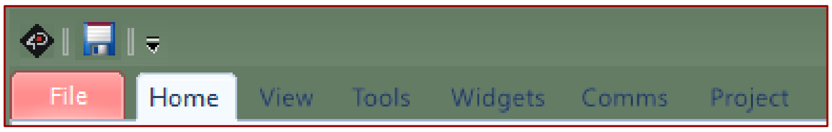
Define the orientation among the four options:



Confirm by clicking on ✓ OK or deny by clicking on ✗ Cancel.

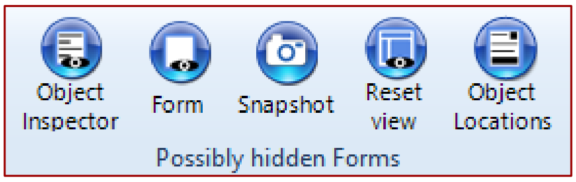
8. ViSi Specific Menus

The ViSi environment includes all the menus available with the Designer environment plus two additional menus: **View** and **Widgets**. The **Project** menu also contains additional selections regarding the destination for Widgets/media.

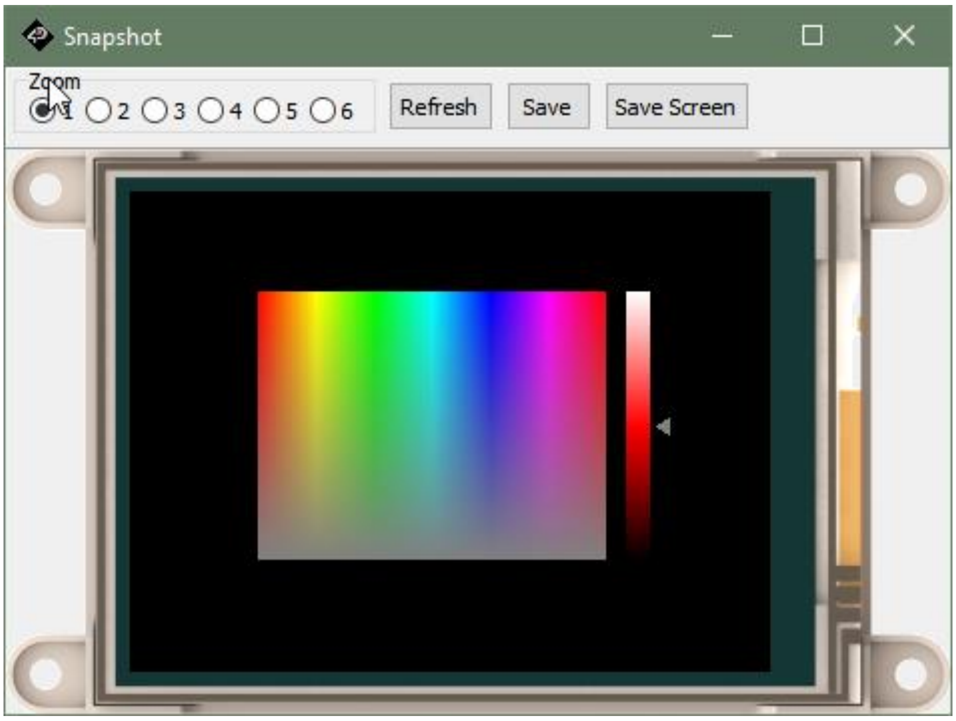


8.1. View Menu

The **View** menu includes one important tool for visualising the form:



Click on **Snapshot** to open a specific window of the form to enable a 1:1 screenshot of the display to me made.

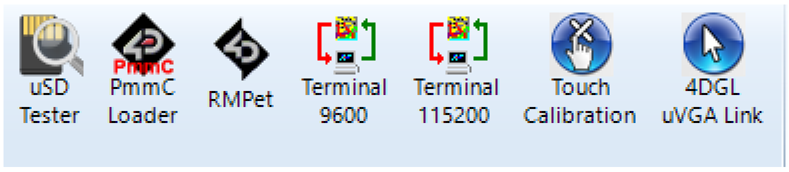


This window provides a zoom up to 4 times. The **Save** button allows to save the screen as an image.

Object Inspector, Form, and Reset View, all relate to the menus and tool bars used in Workshop4. These toolbars and menus can be moved and detached from the side of Workshop4. Object Inspector and Form will bring to front the relevant toolbar when clicked. If required, the toolbars can be reset back to their default location by clicking the Reset View button.

Object Locations enables the user to copy the locations/coordinates of objects on the display, to the clipboard.

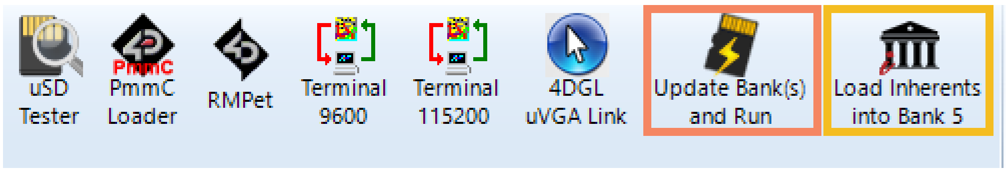
8.2. Tools Menu



All the tools and utilities are grouped here:

- Click **uSD Tester** to upload a test program to the display, to test the inserted uSD card to check compatibility.
- Click **PmmC Loader** to start the PmmC Loader utility.
- Click **RMPet** to start the RMPet utility to partition and format your uSD card, inserted into the PC.
- 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.
- Click **Touch Calibration** to calibrate the touch on the screen. This tool is not available for all display modules.
- Click **4DGL uVGA Link** to open an interactive window to use mouse/keyboard with the uVGA-II or -III module.

For Diablo16 display modules, there is the **Update Bank(s) and Run** icon, and **Load Inherents into Bank 5** icon also.



Update Banks(s) and Run

When **uSD** is selected as the Destination in the Project Menu, the ViSi program will be copied to the uSD card. This option does not require the display module to be connected to the PC during build time. However, this option requires the **Update Bank(s) and Run** program to be downloaded to Bank 0 of the display's flash memory. The **Update Bank(s) and Run** program button is found under the Tools menu.

This **Update Bank(s) and Run** program checks the uSD card for ViSi program files and copies them to their destination flash banks. Then, by default, the program in Bank 1 is executed. The **Update Bank(s) and Run** program can be modified to run the code in another bank besides Bank 1 if desired.

Note that **Update Bank(s) and Run** program stores the time and date information of ViSi program files (for all banks) in Bank 0. Every time that the display module is power cycled, **Update Bank(s) and Run** in Bank 0 always runs first and checks the time and date information of the ViSi program files present in the uSD card. By default, if the time and date information of a ViSi program file is different from that of the last program file uploaded to the same bank, **Update Bank(s) and Run** automatically updates the specified bank.

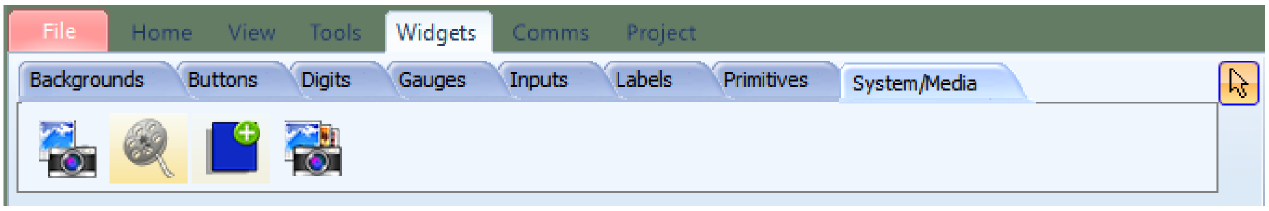
It is also possible to modify the **Update Bank(s) and Run** program such that it only updates the target banks only if the corresponding ViSi program files in the uSD card have a newer time and date information.

Load Inherents into Bank 5

This is a tool which loads the widget software related to **Inherent Widgets** when used on a Diablo16 processor, into Flash Bank 5. This is only required to be done once. All possible Inherent Widget code is loaded in to Bank 5, whether it is used or not, so this is a one-time operation only. It is then ready to go for whatever Inherent widgets you might add to your application. You will receive an error message (Error 30) on the display if you have not loaded Bank 5 based on what you have programmed into the display.

8.3. Widgets Menu

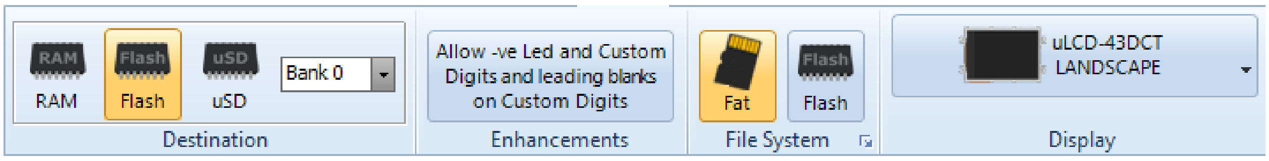
The **Widgets** menu includes the objects pane with all the objects available to build the interface:



The Widgets are detailed in the ***Workshop4 Widgets Reference Manual****, available on the website*.

8.4. Project Menu

The Project menu for ViSi provides additional toggles for the designation of the widgets/media which are used in the Project, as to if they reside in microSD card storage, or Flash memory.

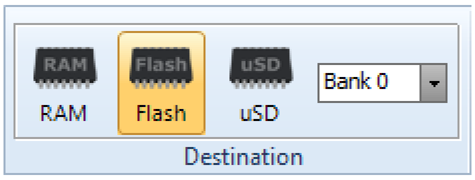


There are four areas in this menu.

- Destination where code will reside,
- Enhancements,
- File System where widgets/media will reside,
- And display selection.

8.4.1. Destination

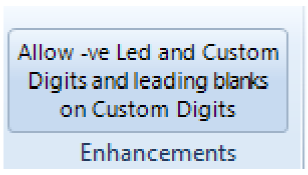
The following options are for selecting the destination of the compiled code (not images/multimedia etc)



Select the **Destination**:

- **RAM** means the display must be connected during build and that the program will be downloaded to the display's RAM memory once compiled. If **RAM** is chosen as the destination, the program is lost when the display is turned off. This really is for testing while developing, to prevent the Flash cycles being used.
- **Flash** means the display must be connected during build and that the program will be downloaded to the display's flash memory once compiled. If **Flash** is chosen as the destination, the program is retained and will be available after power cycling. This should be the default for normal use.
- **uSD** - The user's application will be built and copied to the uSD card. From the uSD card the application is loaded into RAM and run from there. This option requires the **Boot uSD** program (Picaso) or **Update Bank(s) and Run** (Diablo) to be uploaded to the display's flash, as seen in the menu **Tools**. These program loads the user's application from the uSD card at startup and executes it.
- **Bank** – This dropdown simply changes the filename generated when using uSD as Destination, for Diablo16.

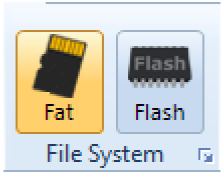
8.4.2. Enhancements


Allow -ve Led and Custom Digits and leading blanks on Custom Digits
Enhancements

The second sections button allows for enabling the use of negative values for LED digits and custom digits objects and for enabling the use of leading blanks on custom digit objects. This is selectable as some older projects were made before this option was enabled, so it is required to be user selected in order to retain historical functionality.

8.4.3. File System

The third section contacts the File System selection, which is to state where the multimedia (widgets, images, etc) will be stored on the display module. Not all display modules have both options, as many will either have a microSD card, or Flash Memory – not both. You will need to select the appropriate one based on your display module.



Selecting Fat will target the multimedia to be stored on the microSD card. Selecting Flash will target the multimedia to be stored on Flash Memory.

Note: Depending on the Widget used in the Project, will determine what type of storage it requires. There are Internal Widgets, Inherent Widgets, and GCI Widgets.

Internal (PmmC) widgets are based in the PmmC and take up code space of the processor (which is the Flash of the processor itself).
Inherent widgets take up External Flash space on Pixxi based modules, or Flash Bank 5 space on Diablo modules. Both also take up a little code space on the Processor. **Note** External Flash is an external chip to the processor, often found on Pixxi based modules. Flash Banks are Diablo specific and are internal to the Diablo processor itself.
GCI widgets take up External Flash Space or microSD space on Pixxi, or microSD space on Diablo.

To clarify, if you have **Internal Widgets** used in your project, they will be stored on the Processor itself alongside your project code, irrespective if you select **Fat** or **Flash** as the File System, as they are stored internal of the processor.
If you have **Inherent Widgets** in your project, if you have the **Fat** option selected for the File System, these widgets are unable to be used as they cannot run from microSD card. If you have **Flash** selected, then these widgets will be stored in External Flash on Pixxi based modules, or in a Flash Bank for Diablo based modules (inside the Diablo processor).
If you have **GCI Widgets** in your project, if **Fat** is selected as the File System, then the widgets will be stored on the microSD card for both Pixxi or Diablo based modules. If **Flash** is selected as the File System, then the widgets will take up External Flash space on Pixxi, and for Diablo it is currently not supported but will be coming soon. **Note** that GCI widgets can become large, and External Flash is limited to typically 16MB for 4D Display modules, so there is a very real chance that GCI widgets may not fit in External Flash, and other widget types such as Internal or Inherent may need to be utilised.

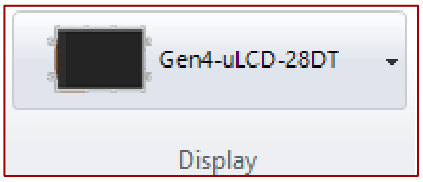
It is important to understand the different widget types BEFORE designing your project. Be sure to know what the intended storage File System is that will be used, and therefore which widget type is the most appropriate for your intended project.

For more information on widgets, all the types and more specific information about them, there is a dedicated document for Widgets available called '**Workshop4 Widgets Reference Manual**'.

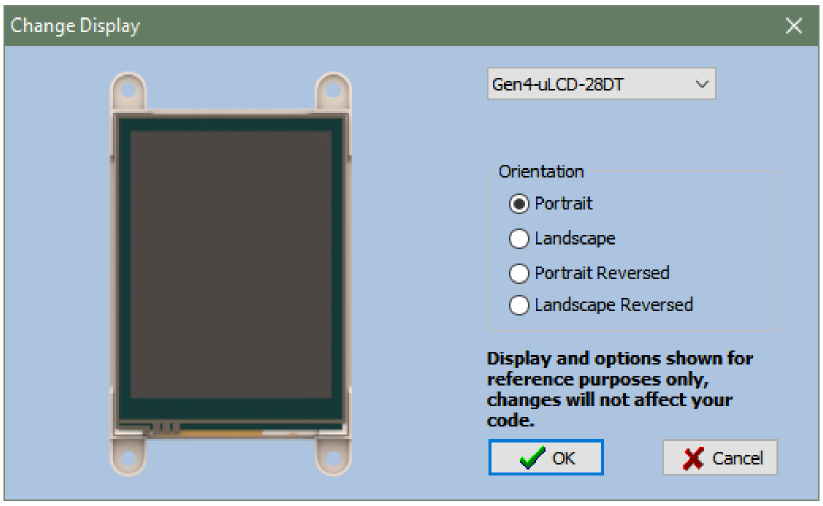
8.4.4. Display Selection

The last section allows selecting the screen, useful for converting a program from one display module to another.

Clicking on the button…

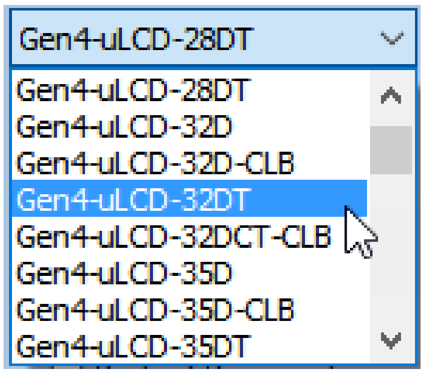


…opens a new window to select the screen:

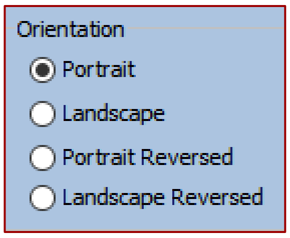


Display and options shown for reference purposes only, changes will not affect your code.

Select the screen from the drop-down list:



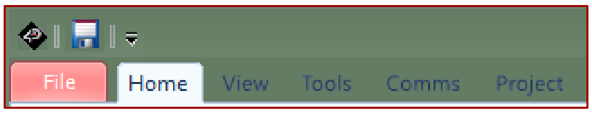
Define the orientation among the four options:



Confirm by clicking on ✓ OK or deny by clicking on ✗ Cancel.

9. ViSi-Genie Specific Menus

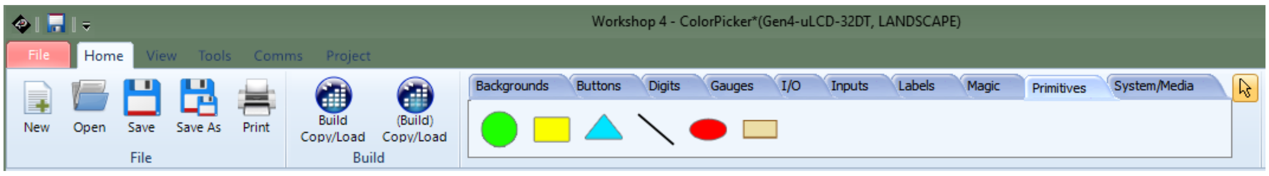
ViSi-Genie includes five menus with specific ribbons and options.



ViSi-Genie is codeless and thus completely different from the previous code-based environments, all the menus relating to ViSi-Genie are detailed.

9.1. Home Menu

The **Home** menu is the main menu.

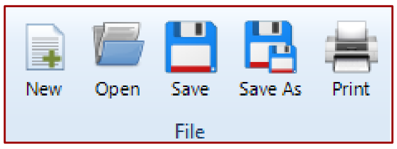


…with three groups of buttons:

- File-related commands,
- Build command,
- And the objects pane.

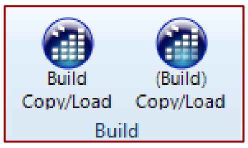
9.1.1. File-Related Buttons

The file-related buttons include the same commands as seen in the File menu: New, Open, Save, Save As and Print.



9.1.2. Build Button

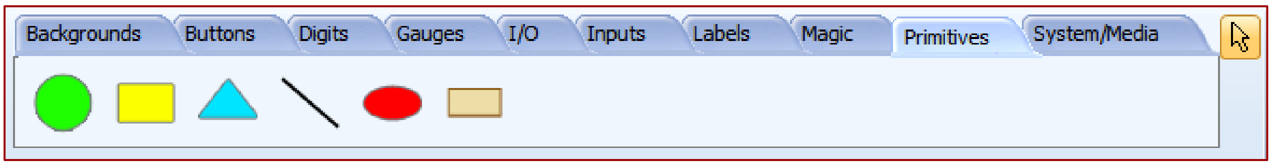
The **Build** button launches the compilation and the generation of the GUI objects, and uploads the project to the display module, and offers to transfer media to uSD/Flash storage. Shortcut Key is Shift + F9
The **(Build)** button will only compile or generate the GUI if required if there have been changes, and then uploads the project to the display module, and offers to transfer media to uSD/Flash storage. Shortcut Key is F9



Additionally, Ctrl + F9 does a forced compile, however this is typically not required in Genie as on the surface it does nothing since the code is all automatically handled in the background. Can be useful if manually editing genie source code however – Advanced users only.

9.1.3. Objects Pane

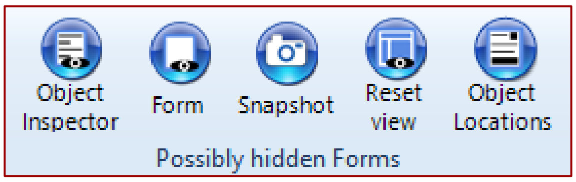
The objects pane includes all the objects available to build the interface:



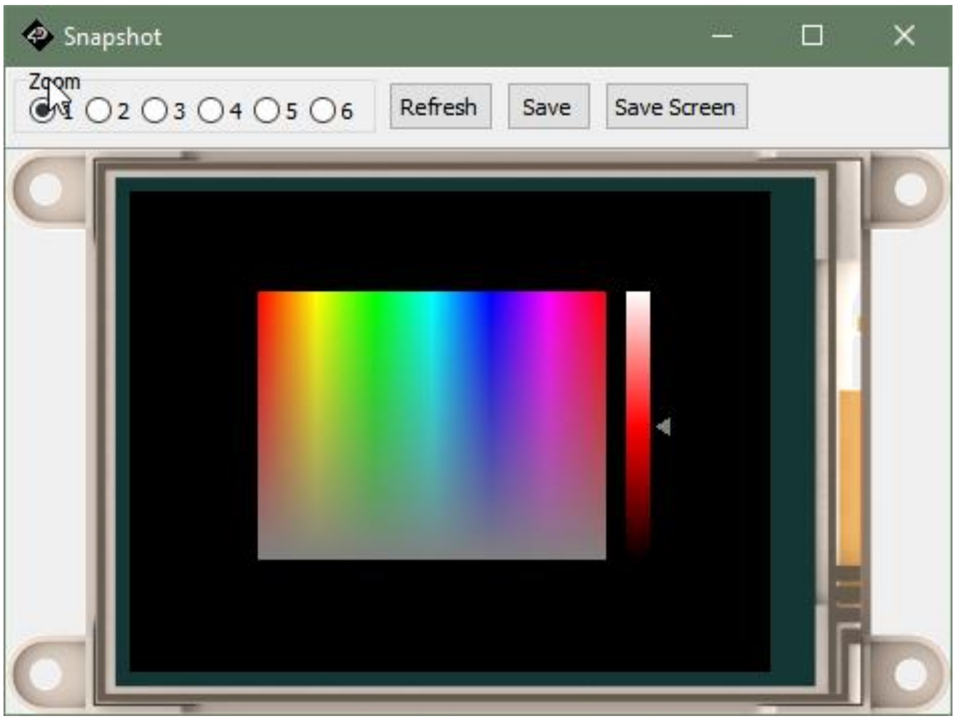
- The objects are detailed on the ***ViSi-Genie User Guide***, available on the website.

9.2. View Menu

The **View** menu includes one important tool for visualising the form:

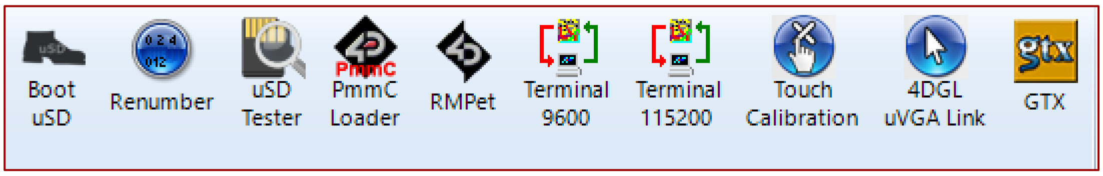


Click on **Snapshot** to open a specific window of the form.



This window provides a zoom up to 6 times. The **Save** button allows to save the screen as an image.

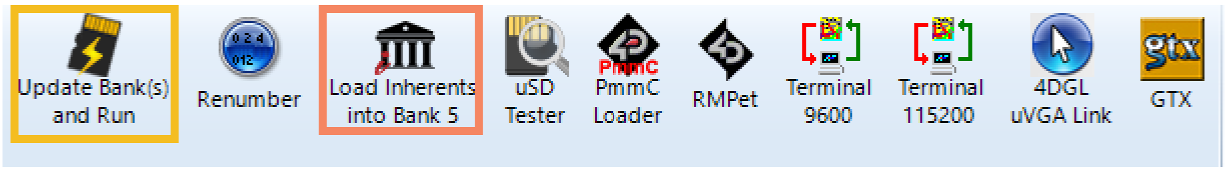
9.3. Tools Menu



All the tools and utilities are grouped here:

- Click **Boot uSD** to upload the Boot uSD application to the screen, enabling programs to be loaded via microSD card. This is Picaso only. See '**Update Bank(s) and Run**' below for Diablo16.
- Click **Renumber** to reallocate the index's of all the widgets in your application, as during development widgets may be deleted, leaving gaps in the numbering index. Optional.
- Click **uSD Tester** to upload a test program to the display, to test the inserted uSD card to check compatibility.
- Click **PmmC Loader** to start the PmmC Loader utility.
- Click **RMPet** to start the RMPet utility to partition and format your uSD card, inserted into the PC.
- 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.
- Click **Touch Calibration** to calibrate the touch on the screen. This tool is not available for all display modules.
- Click **4DGL uVGA Link** to open an interactive window to use mouse/keyboard with the uVGA(GFX) module.
- Click **GTX** to launch the Genie Test Executor debugger.

For Diablo16 display modules, the **Boot uSD** icon is replaced with the **Update Bank(s) and Run** icon, and there is an addition icon called **Load Inherents into Bank 5**.



This **Update Bank(s) and Run** program checks the uSD card for ViSi-Genie program files and copies them to their destination flash banks. Then, by default, the program in Bank 1 is executed. The **Update Bank(s) and Run** program can be modified to run the code in another bank besides Bank 1 if desired.

Note that **Update Bank(s) and Run** program stores the time and date information of ViSi-Genie program files (for all banks) in Bank 0. Every time that the display module is power cycled, **Update Bank(s) and Run** in Bank 0 always runs first and checks the time and date information of the ViSi-Genie program files present in the uSD card. By default, if the time and date information of a ViSi-Genie program file is different from that of the last program file uploaded to the same bank, **Update Bank(s) and Run** automatically updates the specified bank.

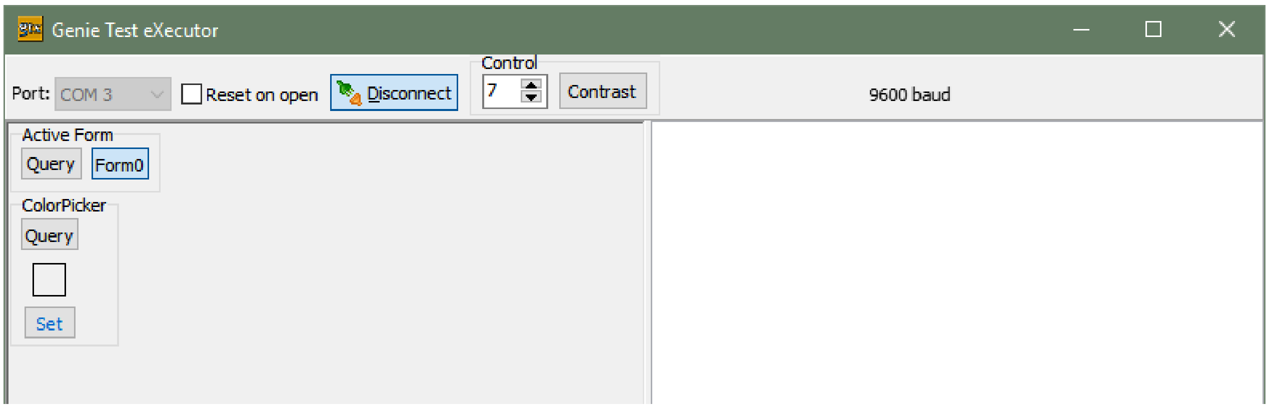
It is also possible to modify the **Update Bank(s) and Run** program such that it only updates the target banks only if the corresponding ViSi-Genie program files in the uSD card have a newer time and date information.

Please refer to the Genie Options section, for more information on the uSD Destination, to utilise this Tool.

Load Inherents into Bank 5, is a tool which loads the widget software related to Inherent Widgets when used on a Diablo16 processor, into Flash Bank 5. This is only required to be done once. All possible Inherent Widget code is loaded in to Bank 5, whether it is used or not, so this is a one-time operation only. It is then ready to go for whatever Inherent widgets you might add to your application. You will receive an error message (Error 30) on the display if you have not loaded Bank 5 based on what you have programmed into the display.

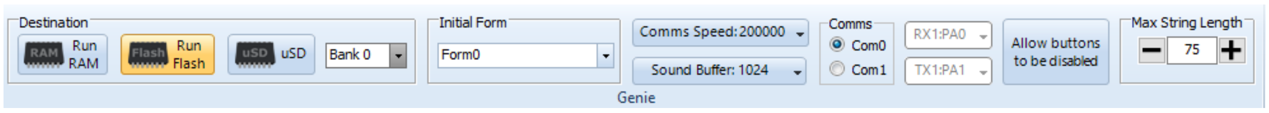
GTX Debugger

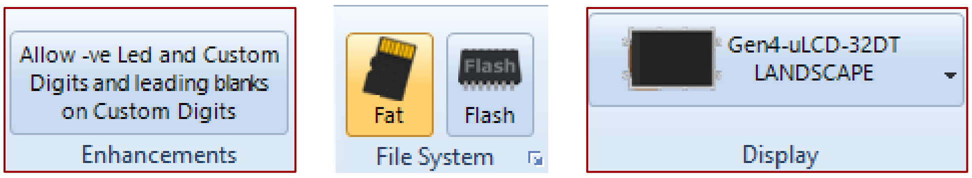
Clicking the GTX button, a new screen will appear with the form and objects defined in the project:



9.4. Project Menu

The **Project** menu includes different parameters and options compared to Designer and ViSi, these will be explained in detail here.



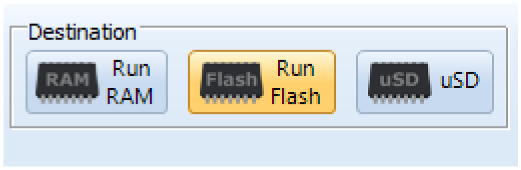

Allow -ve Led and Custom Digits and leading blanks on Custom Digits
Enhancements
Fat
Flash
File System
Gen4-uLCD-32DT LANDSCAPE
Display

There are four groups of buttons:

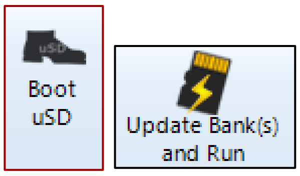
- Options for Genie,
- Enhancements,
- File System,
- And display selection.

9.4.1. Genie Options

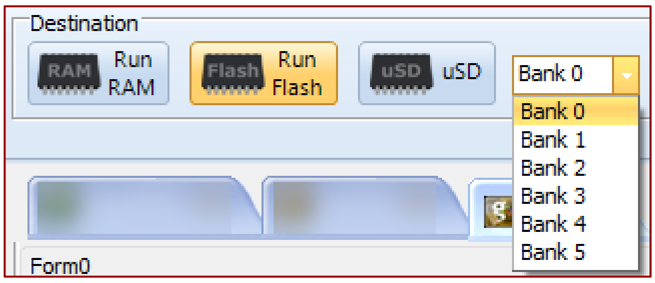
For **Picaso display modules**, the first group includes the Destination options for Genie:



- Select the **Destination** among three possibilities:
  - **Run RAM** - The display must be connected during Build and the program code will be downloaded to the display's flash memory (Processors Flash, not external flash) once compiled. The user's application will be stored in Flash, but will be run from RAM.
  - **Run Flash** - The display must be connected during Build and the program will be downloaded to the display's flash memory (Processors Flash, not external flash) once compiled. The user's application will be stored and run from flash, this uses less memory on the display, but makes programs run slightly slower.
  - **uSD** - The user's application will be built and copied to the uSD card. From the uSD card the application is loaded into RAM and run from there. This option requires the **Boot uSD** program (Picaso) or **Update Bank(s) and Run** (Diablo) to be uploaded to the display's flash, as seen in the menu **Tools**. These program loads the user's application from the uSD card at startup, and executes it.



For **Diablo16 display modules**, the first group contains the designation options shown below, with the addition of selectable Flash Banks – which Picaso does not have.

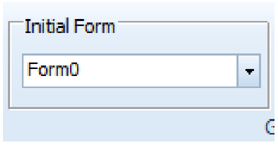


The additional drop-down menu allows the user to specify the target destination flash bank of the ViSi-Genie program. The Diablo16 processor has six flash banks (Bank 0 to Bank 5), each of which has a capacity of 32 kB.

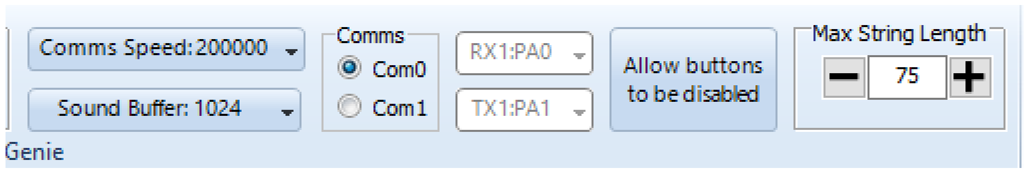
When **Run Flash** is selected, the destination of the ViSi-Genie program is the bank specified in the drop-down menu. In this case, the display module needs to be connected to the PC during build time. The program will then be downloaded to the selected bank and it will run from there. Take note however, that, after the display module is power cycled, the program in Bank 0 always runs first.

On the other hand, when **uSD** is selected, the ViSi-Genie program will be copied to the uSD card. This option does not require the display module to be connected to the PC during build time. However, this option requires the **Update Bank(s) and Run** program to be downloaded to Bank 0 of the display's flash memory. The **Update Bank(s) and Run** program button is found under the Tools menu. See the Tools Menu section for more information.

The initial form section allows the user to set which form will show upon boot up.

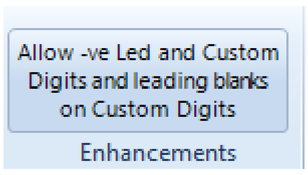


The next part of the Genie options includes advanced parameters:



- **Comms speed** is the baud rate at which the serial command interface operates for this specific project. This speed OVERRIDES the default settings found in the Genie tab of the Options page (which are the Default settings for newly created projects only). Changing the comms speed baud rate here will not affect any other project and is the comms speed which is compiled and loaded into the display module.
- Define **Sound buffer size** to set aside RAM for buffering wav (sound) files. For simple sound files 1024 bytes should be enough. For complicate sound files to be played whilst video is displaying may need as much as 4096 bytes. These settings override the settings found in the Genie Tab of the Options page (which are the Default settings for newly created projects only). Changing the Sound Buffer Size here will not affect any other project and is the buffer size which is compiled and loaded into the display module.
- The **Comms** port selection allows you to change the project specific options for this single project, either utilising the default COM0 for Genie Comms to the host, or COM1 which can have the RX/TX GPIO selected. These settings override the settings found in the Genie Tab of the Options page (which are the Default settings for newly created projects only). Changing the comms port here will not affect any other project and is the comms port which is compiled and loaded into the display module.
- Button objects can be shown and hidden accordingly by the host controller. To enable this, click on the "**Allow buttons to be disabled**" button. Then use the GTX tool to see the appropriate commands for enabling and disabling the buttons.
- **Max String Length** allows the adjustment of the maximum string length able to be sent over Genie comms from the Host to the display. This may be desired to change if a longer string is required to be sent, however doing so will consume an extra 3 bytes of RAM for each additional character. This is Project specific only, the default length for new projects is adjusted in the Options – Genie tab.

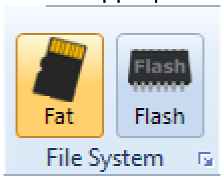
9.4.2. Enhancements


Allow -ve Led and Custom Digits and leading blanks on Custom Digits
Enhancements

This button allows for enabling the use of negative values for LED digits and custom digits objects and for enabling the use of leading blanks on custom digit objects. This is selectable as some older projects were made before this option was enabled, so it is required to be user selected in order to retain historical functionality.

9.4.3. File System

The third section contacts the File System selection, which is to state where the multimedia (widgets, images, etc) will be stored on the display module. Not all display modules have both options, as many will either have a microSD card, or Flash Memory – not both. You will need to select the appropriate one based on your display module.



Selecting Fat will target the multimedia to be stored on the microSD card. Selecting Flash will target the multimedia to be stored on Flash Memory.

Note: Depending on the Widget used in the Project, will determine what type of storage it requires. There are Internal Widgets, Inherent Widgets, and GCI Widgets.

Internal (PmmC) widgets are based in the PmmC and take up code space of the processor (which is the Flash of the processor itself).
Inherent widgets take up External Flash space on Pixxi based modules, or Flash Bank 5 space on Diablo modules. Both also take up a little code space on the Processor. **Note** External Flash is an external chip to the processor, often found on Pixxi based modules. Flash Banks are Diablo specific and are internal to the Diablo processor itself.
GCI widgets take up External Flash Space or microSD space on Pixxi, or microSD space on Diablo.

To clarify, if you have **Internal Widgets** used in your project, they will be stored on the Processor itself alongside your project code, irrespective if you select **Fat** or **Flash** as the File System, as they are stored internal of the processor.
If you have **Inherent Widgets** in your project, if you have the **Fat** option selected for the File System, these widgets are unable to be used as they cannot run from microSD card. If you have **Flash** selected, then these widgets will be stored in External Flash on Pixxi based modules, or in a Flash Bank for Diablo based modules (inside the Diablo processor).
If you have **GCI Widgets** in your project, if **Fat** is selected as the File System, then the widgets will be stored on the microSD card for both Pixxi or Diablo based modules. If **Flash** is selected as the File System, then the widgets will take up External Flash space on Pixxi, and for Diablo it is currently not supported but will be coming soon. **Note** that GCI widgets can become large, and External Flash is limited to typically 16MB for 4D Display modules, so there is a very real chance that GCI widgets may not fit in External Flash, and other widget types such as Internal or Inherent may need to be utilised.

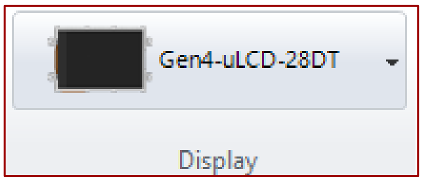
It is important to understand the different widget types BEFORE designing your project. Be sure to know what the intended storage File System is that will be used, and therefore which widget type is the most appropriate for your intended project.

For more information on widgets, all the types and more specific information about them, there is a dedicated document for Widgets available called '**Workshop4 Widgets Reference Manual**'.

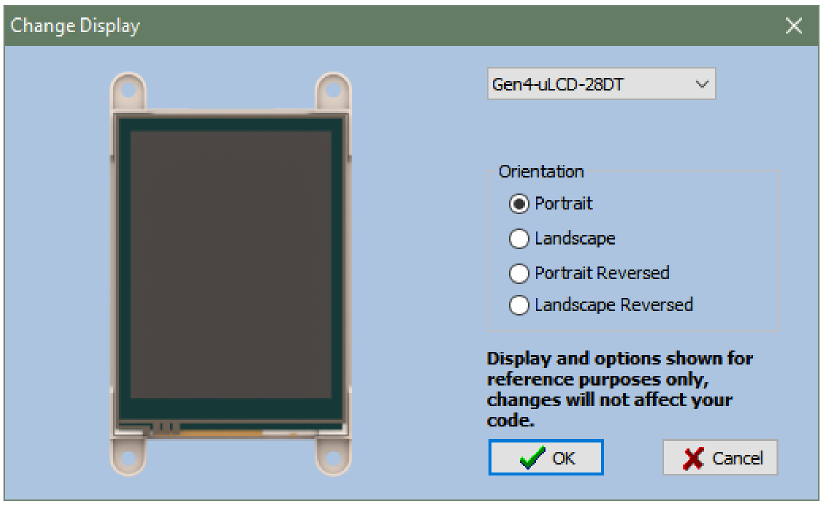
9.4.4. Display Selection

The last section allows selecting the screen, useful for converting a program from one display module to another.

Clicking on the button…

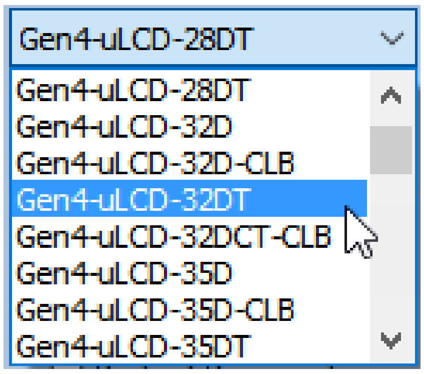


…opens a new window to select the screen:

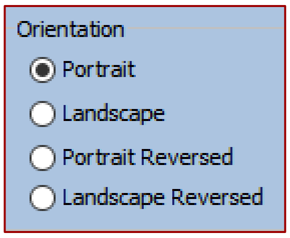


Display and options shown for reference purposes only, changes will not affect your code.

Select the screen from the drop-down list:



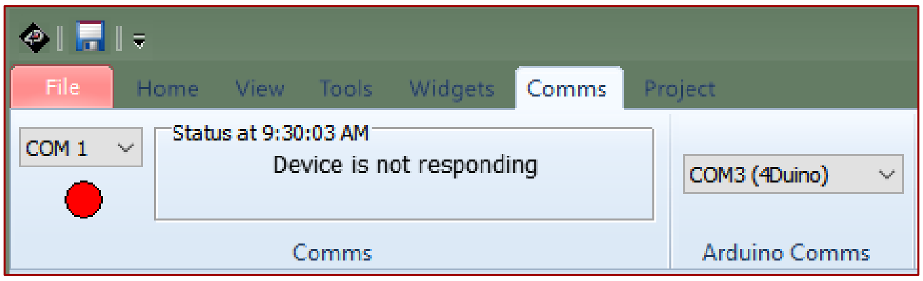
Define the orientation among the four options:



Confirm by clicking on ✓ OK or deny by clicking on ✗ Cancel.

10. Basic and Extended Graphics Specific Menus

The Arduino compatible Basic and Extended Graphics environments include all the menus available with the ViSi Environment with some additional options for **Arduino Comms** which can be found under **Comms** tab.



Status at 9:30:03 AM
Device is not responding

COM3 (4Duino)

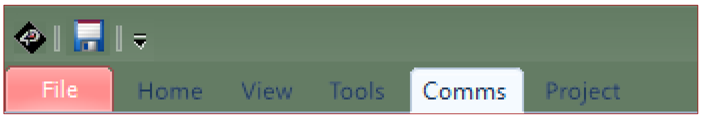
The Arduino Comms refers to the COM port that the Arduino board is currently using.

The main difference between the Basic and Extended Arduino environments lies on the available widgets.

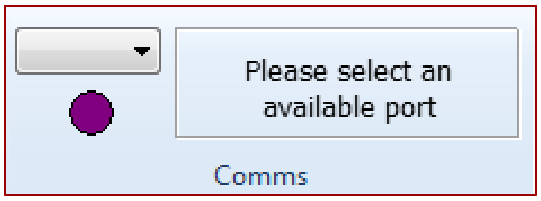
Since the Basic Environment is designed to allow users to create projects without the need for a uSD card, it only allows the user to use primitive shapes and objects in the WYSIWYG window. The Extended Graphics on the other hand gives additional support for 4D Graphics.

11. Connect the Module

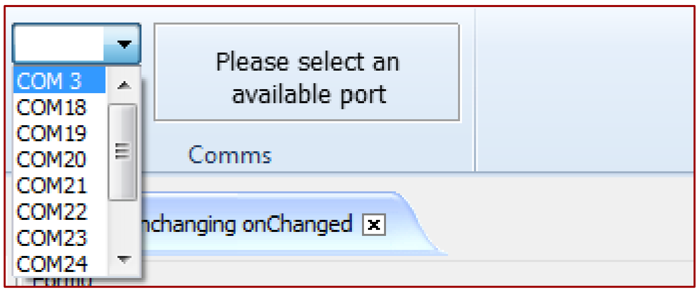
Connect the module to a USB port with the 4D Systems programming cable and select the **Comms** menu:



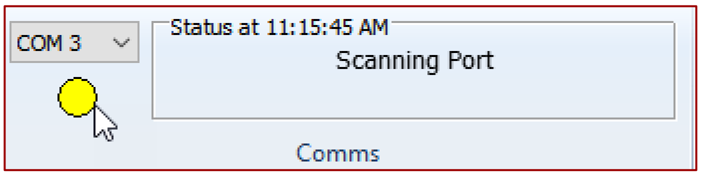
Above the **Comms** section, the **violet** light mentions no module is currently connected.



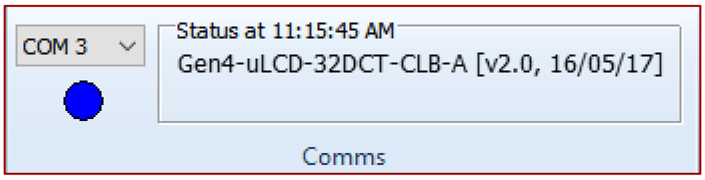
Connect the 4D Systems programming cable/adaptor to the module and plug the cable into the USB port. Click on the drop-down list and select the COM port relating to the 4D Programming cable/adaptor.



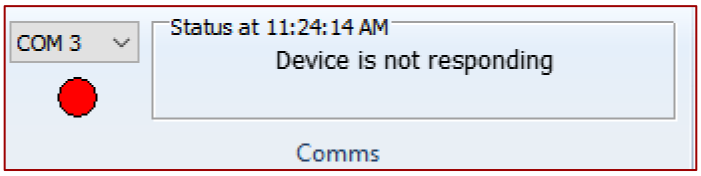
The light turns yellow while the connection is being established:



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



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

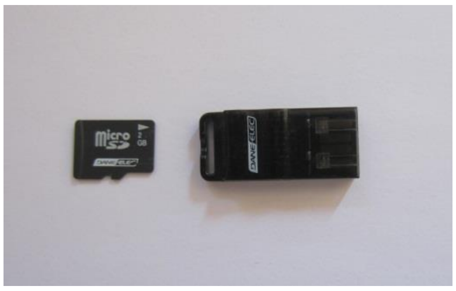


12. Insert the Micro-SD Card

For Picaso, Pixxi-28, Pixxi-44 and Diablo16, the micro-SD card shall be FAT16-formatted. Partition can't exceed 4 GB. For Goldelox, the micro-SD card shall not be formatted at all, it requires the SD card to be RAW.

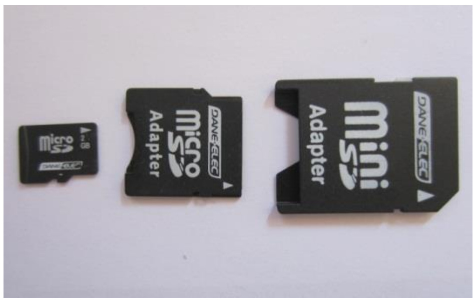
To connect the micro-SD card, either:

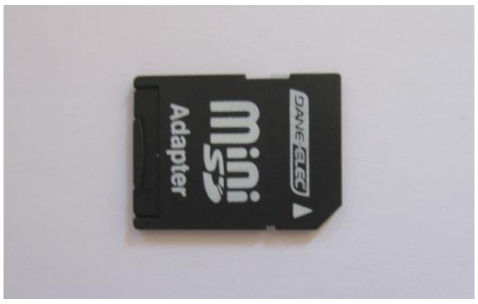
- Insert the micro-SD card into the USB adaptor and plug the USB adaptor into an USB port of the PC.



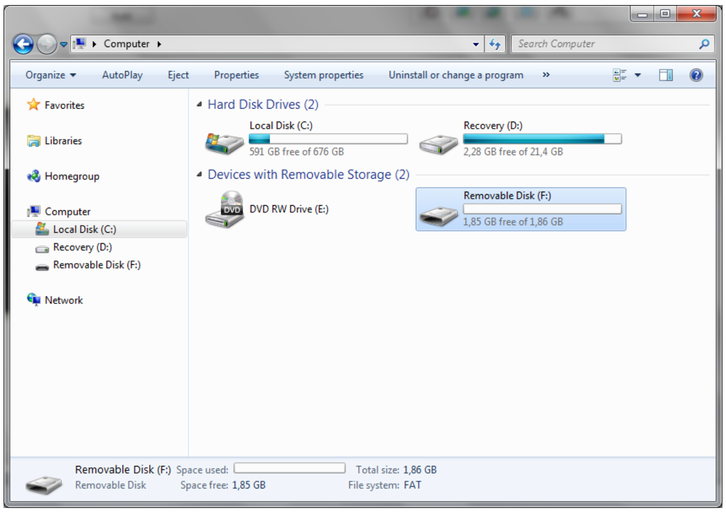
Or

- 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 F:.



Computer
Search Computer
Organize AutoPlay Eject Properties System properties Uninstall or change a program

Favorites
Libraries
Homegroup
Computer
Local Disk (C:)
Recovery (D:)
Removable Disk (F:)
Network

Hard Disk Drives (2)
Local Disk (C:) 591 GB free of 676 GB
Recovery (D:) 2,28 GB free of 21,4 GB

Devices with Removable Storage (2)
DVD RW Drive (E:)
Removable Disk (F:) 1,85 GB free of 1,86 GB

Removable Disk (F:) Space used: Total size: 1,86 GB
Removable Disk Space free: 1,85 GB File system: FAT

It is highly recommended to use the Workshop4 Tool called **RMPet** when formatting and partitioning your microSD card in your PC, for use in 4D Systems modules.

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

13. Revision History

Revision	Revision Content	Revision Date
1.0	First Release	19/11/2012
1.1	Typos on Page 4 fixed	17/12/2012
1.2	Added new content for Serial and fixed incorrect document references	04/02/2013
1.3	Amended details about Micro-SD card	05/07/2013
1.4	Amended details about Program Destination	11/03/2014
2.0	Updated formatting and contents	01/052017
2.1	Added information on target flash banks	29/07/2017
2.2	Updated Formatting	05/04/2019
2.3	Added more information around Project Options vs Options Menu, regarding default settings vs Project specific settings. Added information regarding Fat/Flash designations for widgets and media. Added ViSi options for Update Flash Banks() and Run for Diablo.	27/07/2020

14. Legal Notice

15. Contact Information

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