Workshop4 User Manual

Manual

Revision 2.5

Copyright © 2024 4D Systems

Content may change at any time. Please refer to the resource centre for latest documentation.

Contents

1. Introduction 5
   1.1. Requirements 6
   1.2. Installation 7
   1.3. Programming Tools Driver Installation 11
      1.3.1. CP210x VCP Driver 11
      1.3.2. FTDI VCP Driver 13
   1.4. MicroSD Card Format 15
2. Creating a New Project 17
3. 4D Environments 19
   3.1. Designer 19
   3.2. ViSi 20
   3.3. ViSi-Genie 21
   3.4. Serial 23
   3.5. Additional Options 25
      3.5.1. Create System File 25
      3.5.2. Create Text File 26
4. Arduino Compatible Environments 27
   4.1. Basic Graphics 28
   4.2. Extended Graphics 28
   4.3. Genie Graphics 28
5. Common File Menu 29
6. Designer Specific Menus 31
   6.1. Home Menu 31
      6.1.1. File-Related Buttons 31
      6.1.2. Code-Related Buttons 31
      6.1.3. Bookmark Buttons 32
      6.1.4. Find and Replace Buttons 32
      6.1.5. Code Folding Buttons 33

6.1.6. Compile Buttons 34
6.2. Tools Menu 35
6.3. Comms Menu 36
6.4. Project Menu 36
6.4.1. Destination 36
6.4.2. Enhancements 37
6.4.3. Display Selection 37
7. ViSi Specific Menus 38
7.1. View Menu 38
7.2. Tools Menu 39
7.3. Widgets Menu 40
7.4. Project Menu 41
7.4.1. Destination 41
7.4.2. Enhancements 42
7.4.3. File System 42
7.4.4. Display Selection 43
8. ViSi-Genie Specific Menus 44
8.1. Home Menu 44
8.1.1. File-Related Buttons 44
8.1.2. Build Buttons 45
8.1.3. Objects Pane 45
8.2. View Menu 46
8.3. Tools Menu 47
8.4. Project Menu 49
8.4.1. Genie Options 49
8.4.2. Enhancements 51
8.4.3. File System 52
8.4.4. Display Selection 52
9. Basic/Extended Graphics Specific Menus 53
10. Connect the Module 54

11. Workshop4 Widgets 55
11.1. Smart Widgets 55
12. Revision History 56
13. Legal Notice 57
13.1. Proprietary Information 57
13.2. Disclaimer of Warranties & Limitations of Liabilities 57

1. Introduction

This user guide introduces 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 Arduino Compatible Environments section.

To install Workshop4, please refer to Installation section.

1.1. Requirements

The following are required for the installation and use of Workshop4:

- **Windows 8 or newer**
- Workshop4 IDE Installer

Some older Windows OS's such as ME, Vista, XP and 7 have not been tested for some time and are not supported by Microsoft anymore, however, may still work.

Legacy support updates for the older OS's may not be possible, depending what issues are found. For best results, run a current Windows OS.

> **Note**
>
> It is also possible to run Workshop4 under a VM on Linux or Mac, however it is up to the user to set this up.

Listed are tools recommended during development:

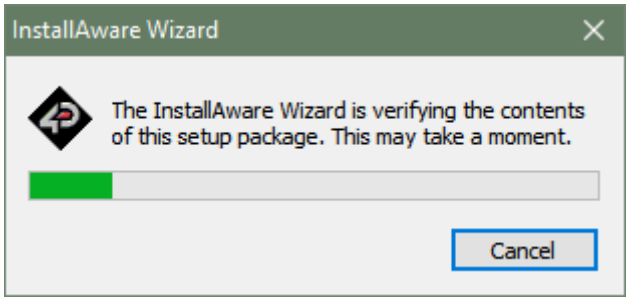
- 4D-UPA
- gen4-PA
- uUSB-PA5 (*w/ gen4-IB if using gen4 and pixxiLCD displays*)
- 4D Programming Cable (*w/ gen4-IB if using gen4 and pixxiLCD displays*)

All are available from **4D Systems**. These hardware tools are used to update the PmmC and upload Workshop4 projects to the graphics processors.

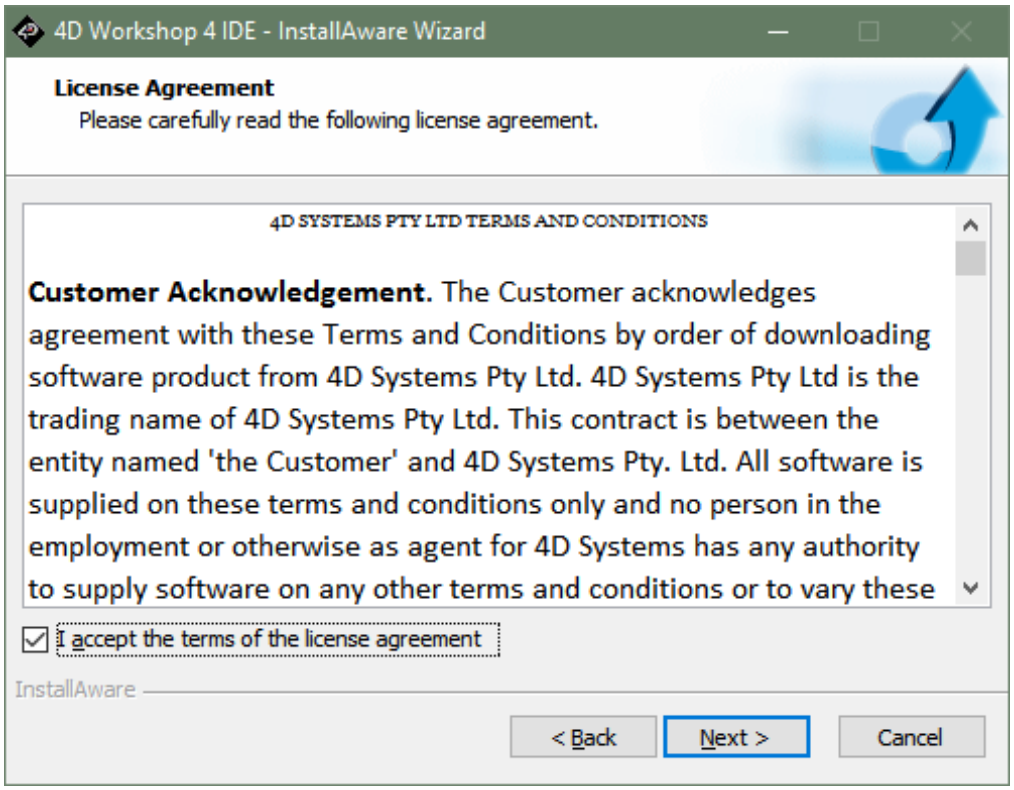
1.2. Installation

After acquiring a copy of the installer, you should be able to run it simply by double clicking the file.

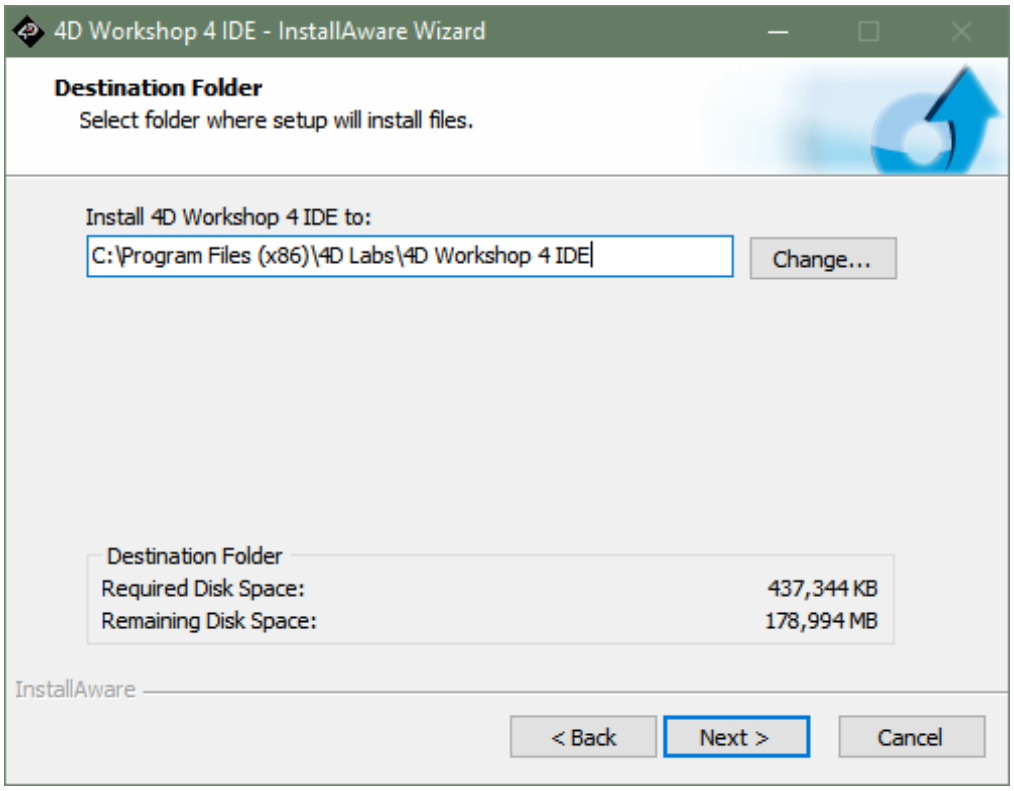
The installation starts by verifying the contents of the setup package.



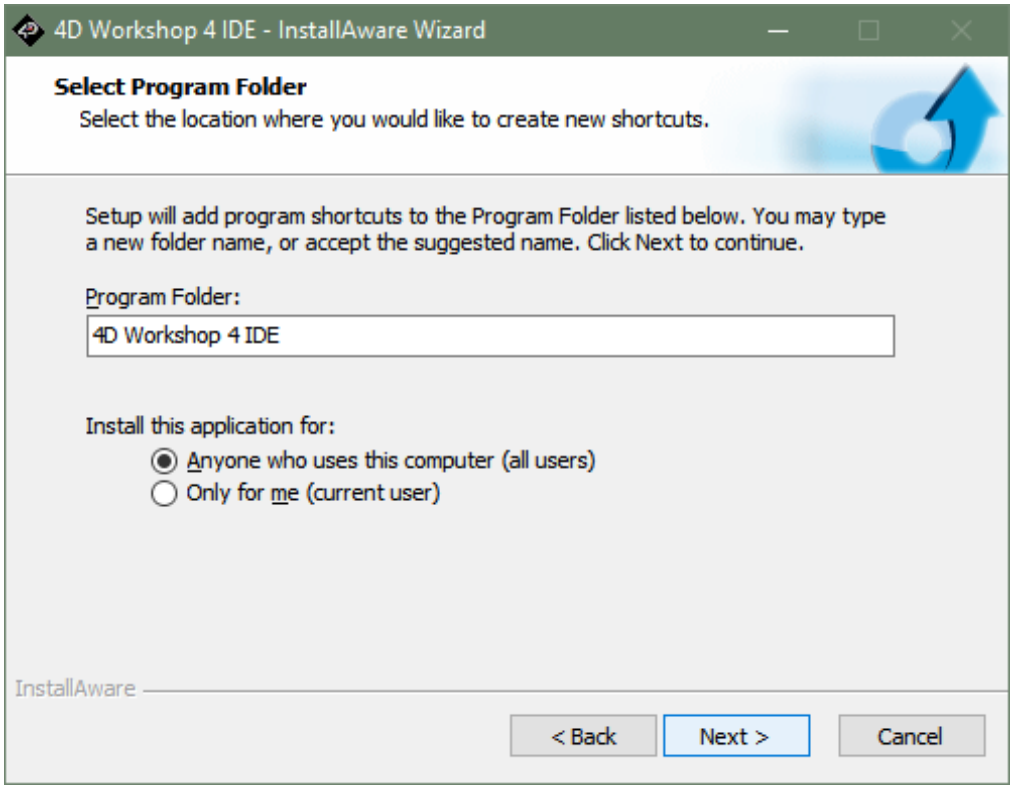
Once done, the installer will open the Terms and Conditions window. Please read the agreement carefully then tick the checkbox and click on **Next** to continue.



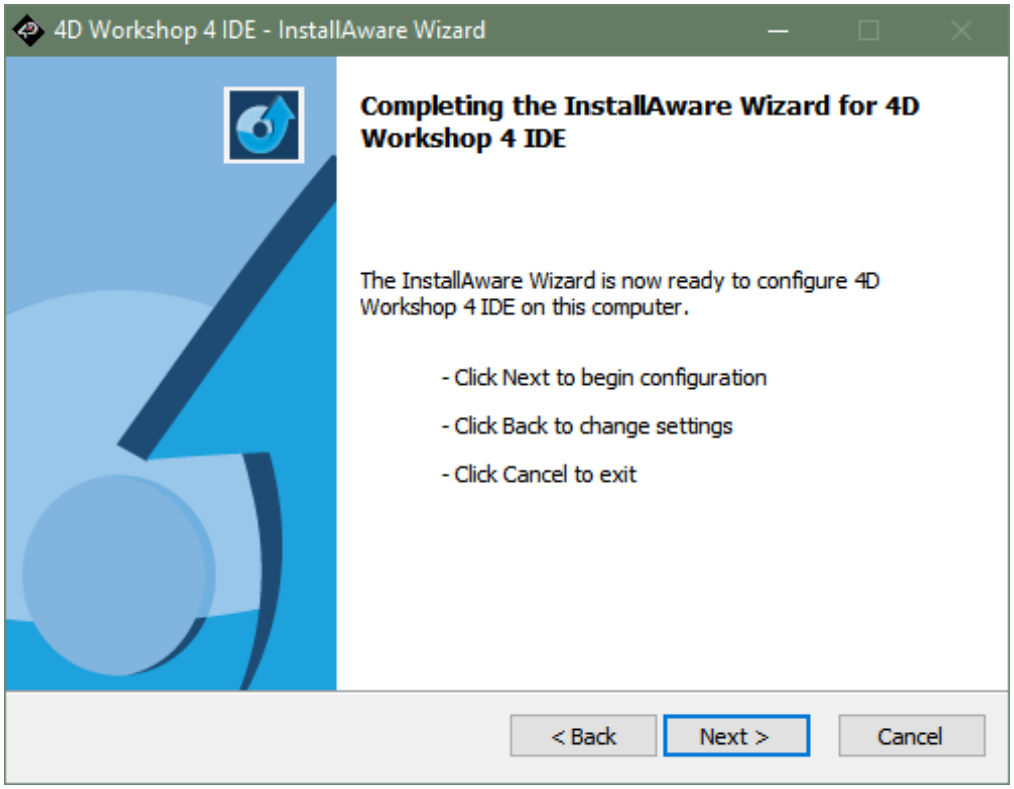
The installer will then let you select the installation directory. You can skip this step and simply click on **Next**.



You have an option to install the application for all users or only for the current user, hence yourself. Simply click on **Next** to continue.



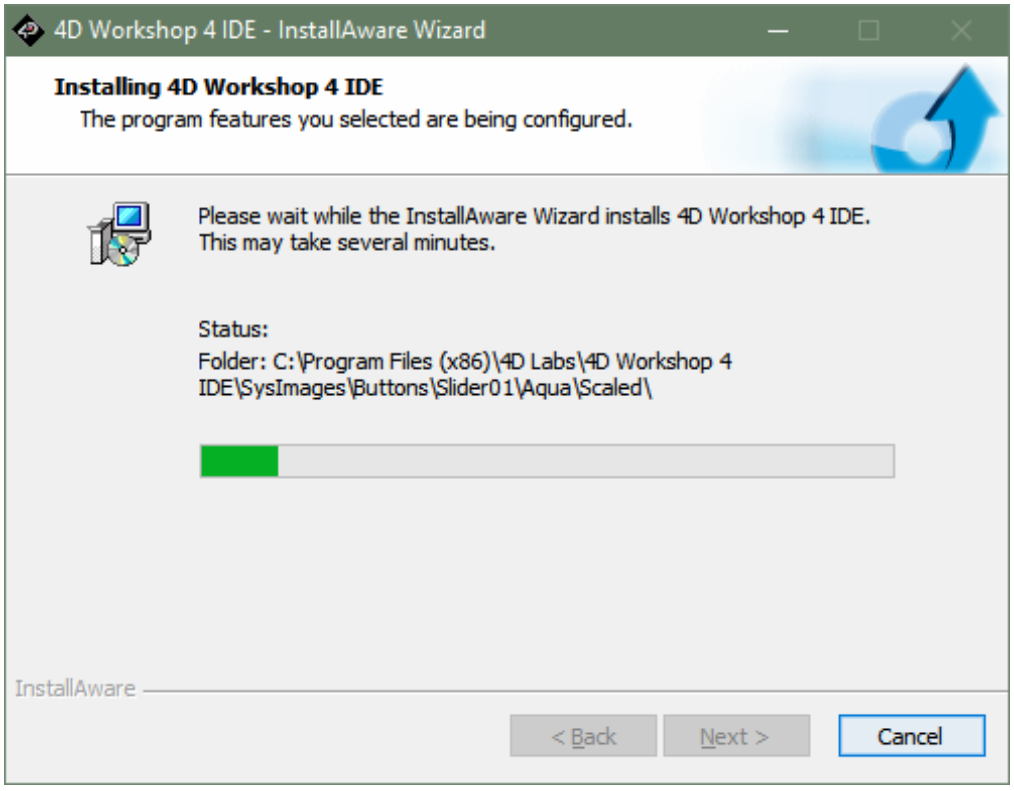
If all is set and done to your liking simply continue from this screen by clicking on **Next**. Otherwise, click on **Back** to change what you've set.



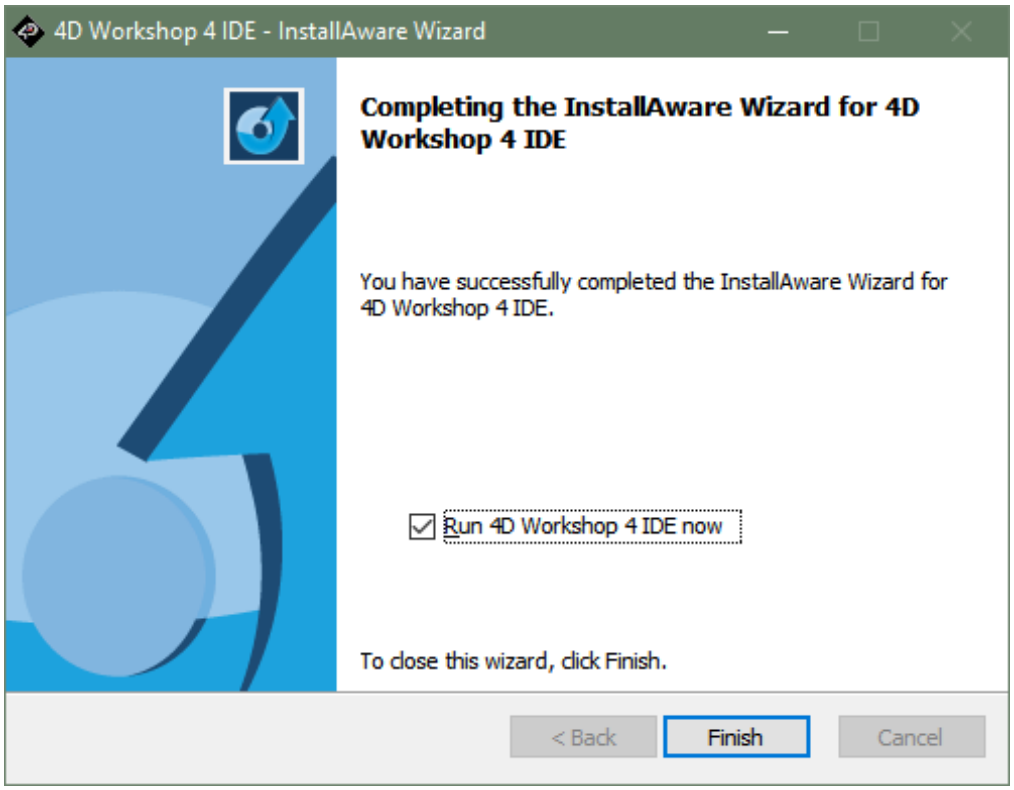
Note

It is recommended to stick to the default configurations.

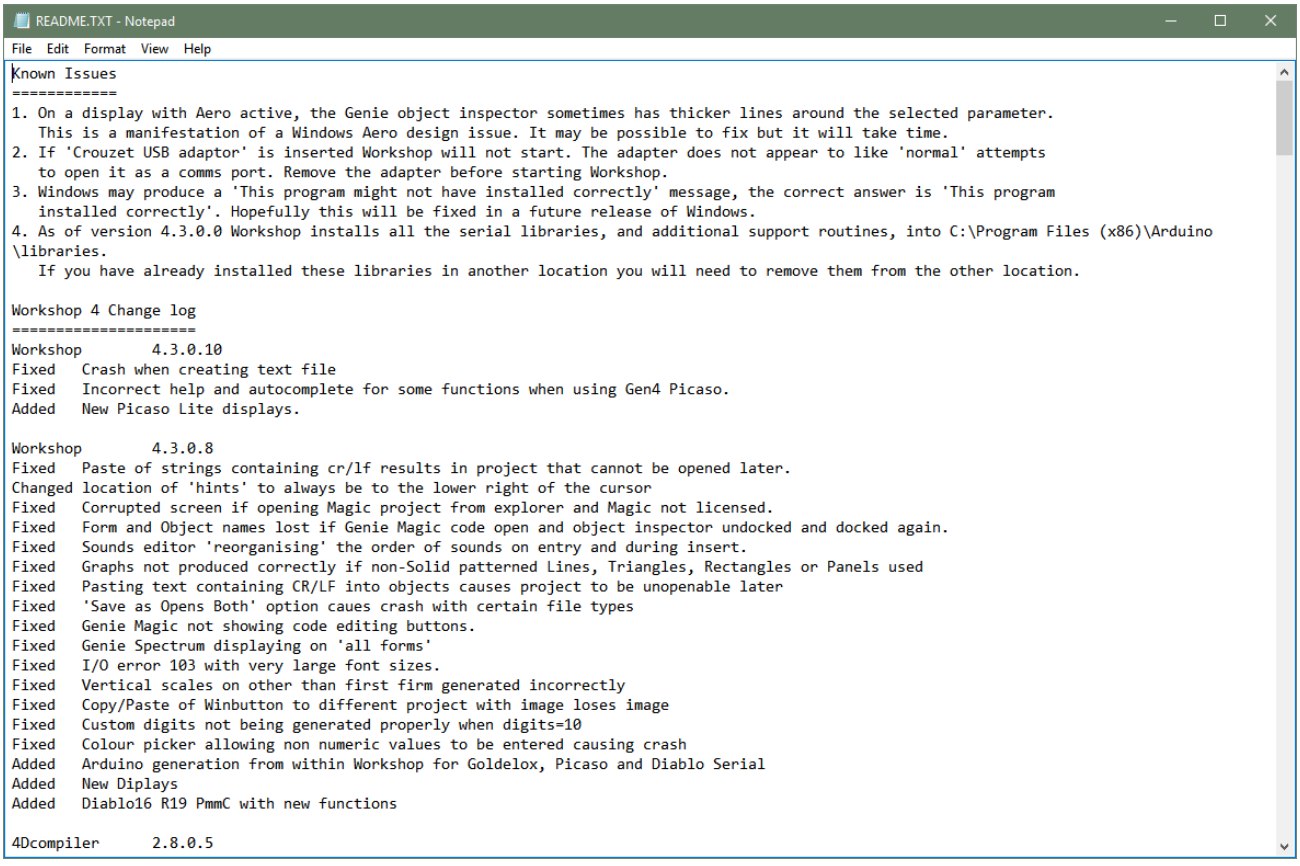
Simply wait for the installation to finish.



After the installation, you are given an option to run Workshop4 IDE after closing the installation window. If you want to do so, leave a check mark on the checkbox. Click on **Finish**.



Besides the last window from the installer, a text editor session will open a **README.TXT** file. This file contains the change log and known issues.



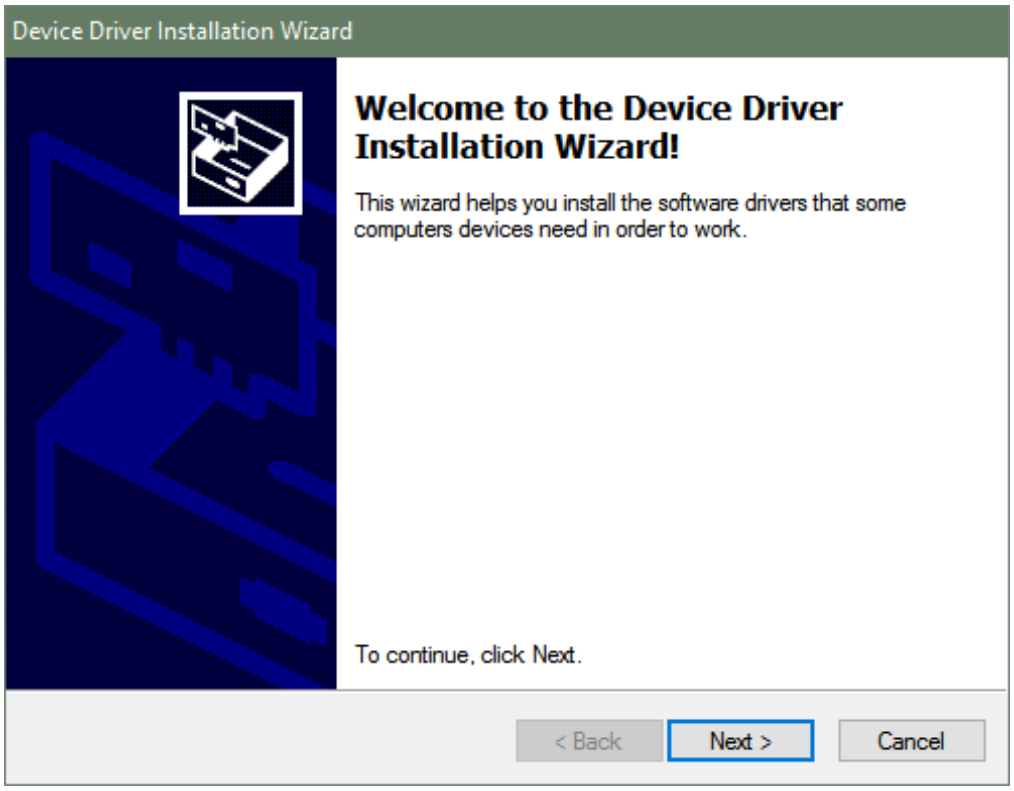
1.3. Programming Tools Driver Installation

You can find the links of the drivers for each recommended USB to TTL programming solutions on their product pages.

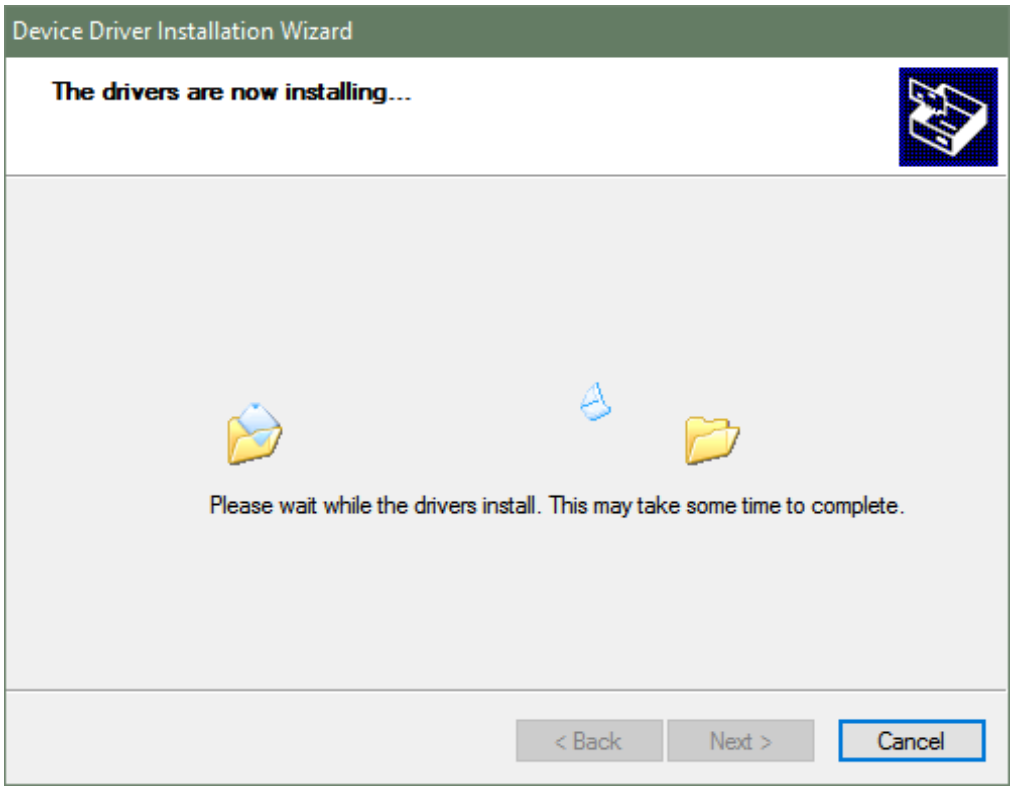
1.3.1. CP210x VCP Driver

This is used for **4D Programming Cable**, **uUSB-PA5-II**, **gen4-PA** and **4D-UPA**. This driver can be downloaded here.

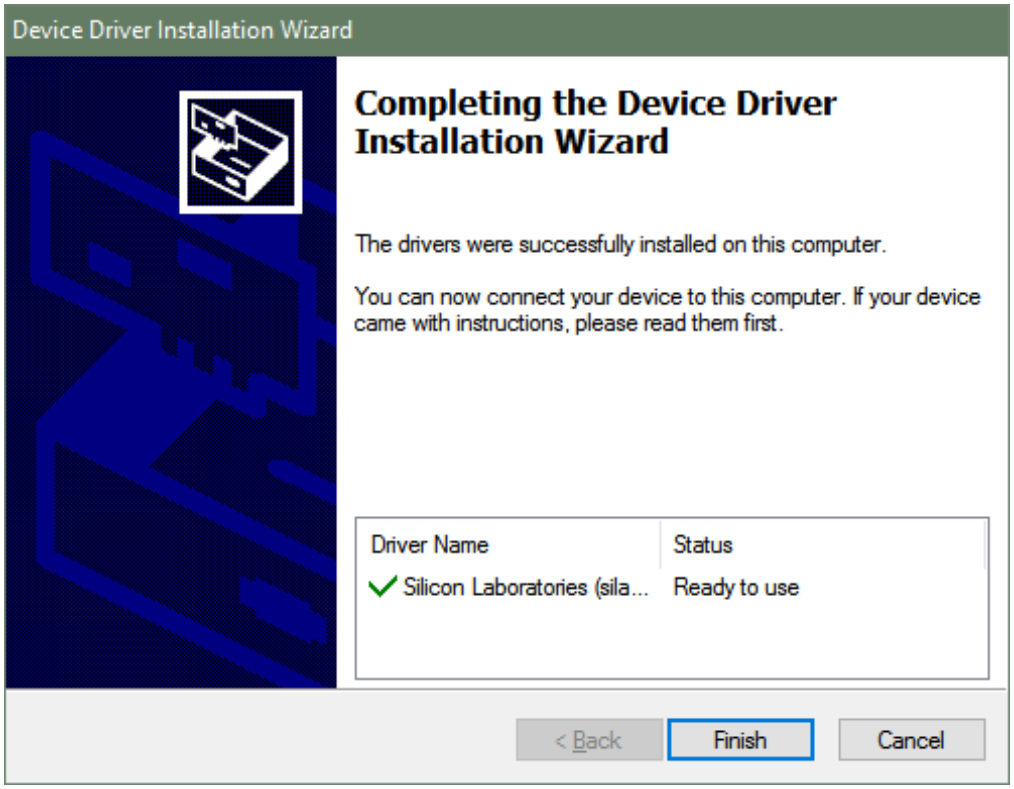
After downloading successfully, save the zipped file into a folder on your system and unzip the file. Launch the appropriate installer based on your computer's architecture.



Continue with the installation prompts by clicking on **Next**.



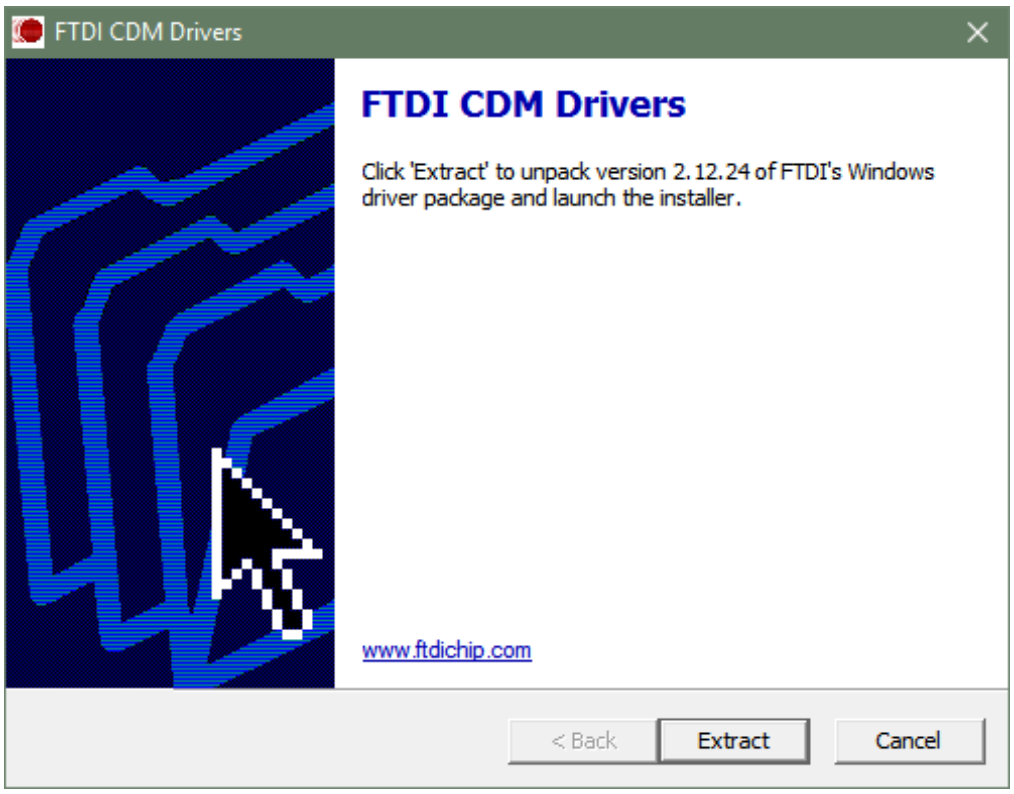
The installation should finish shortly.



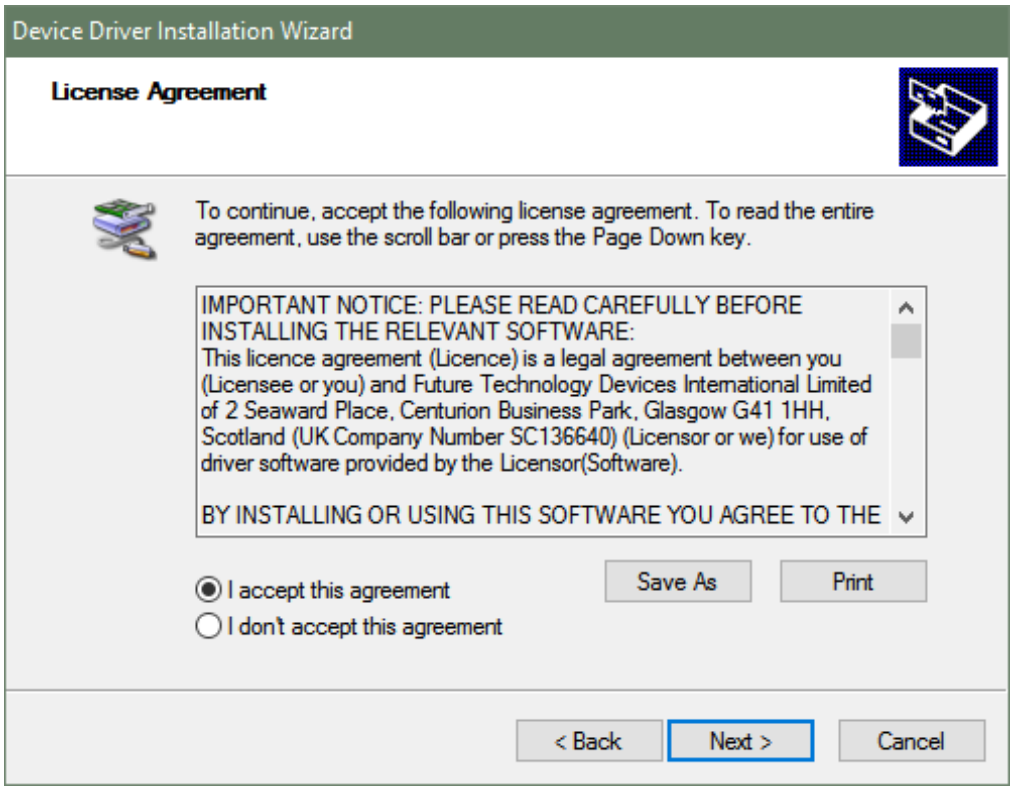
1.3.2. FTDI VCP Driver

The installer for this driver can be found here. This driver is used solely for uUSB-PA5 which has been superseded by the uUSB-PA5-II.

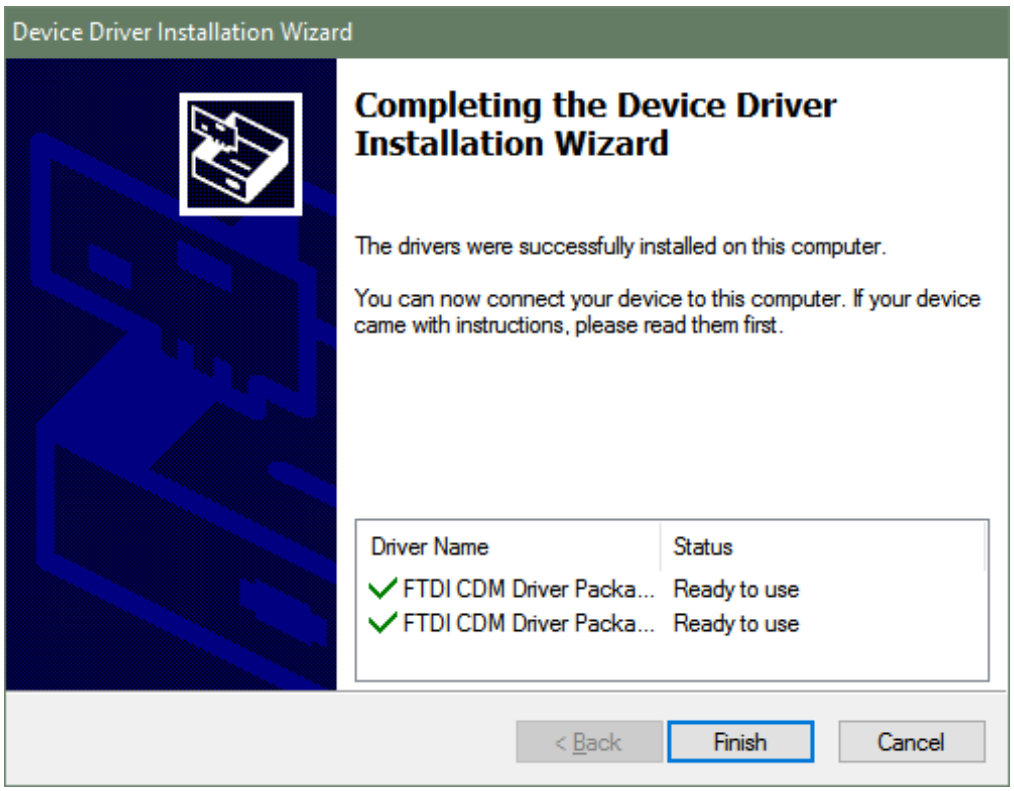
After downloading, save the zipped file into a folder on your system and unzip the file. Launch the installer afterwards.



Accept the agreement and continue with the installation prompts by clicking on Next.

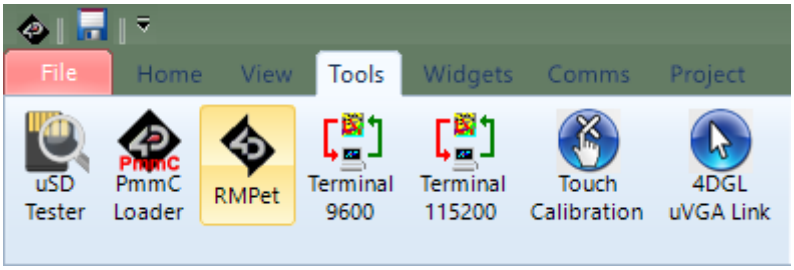


The installation should finish quickly afterwards.

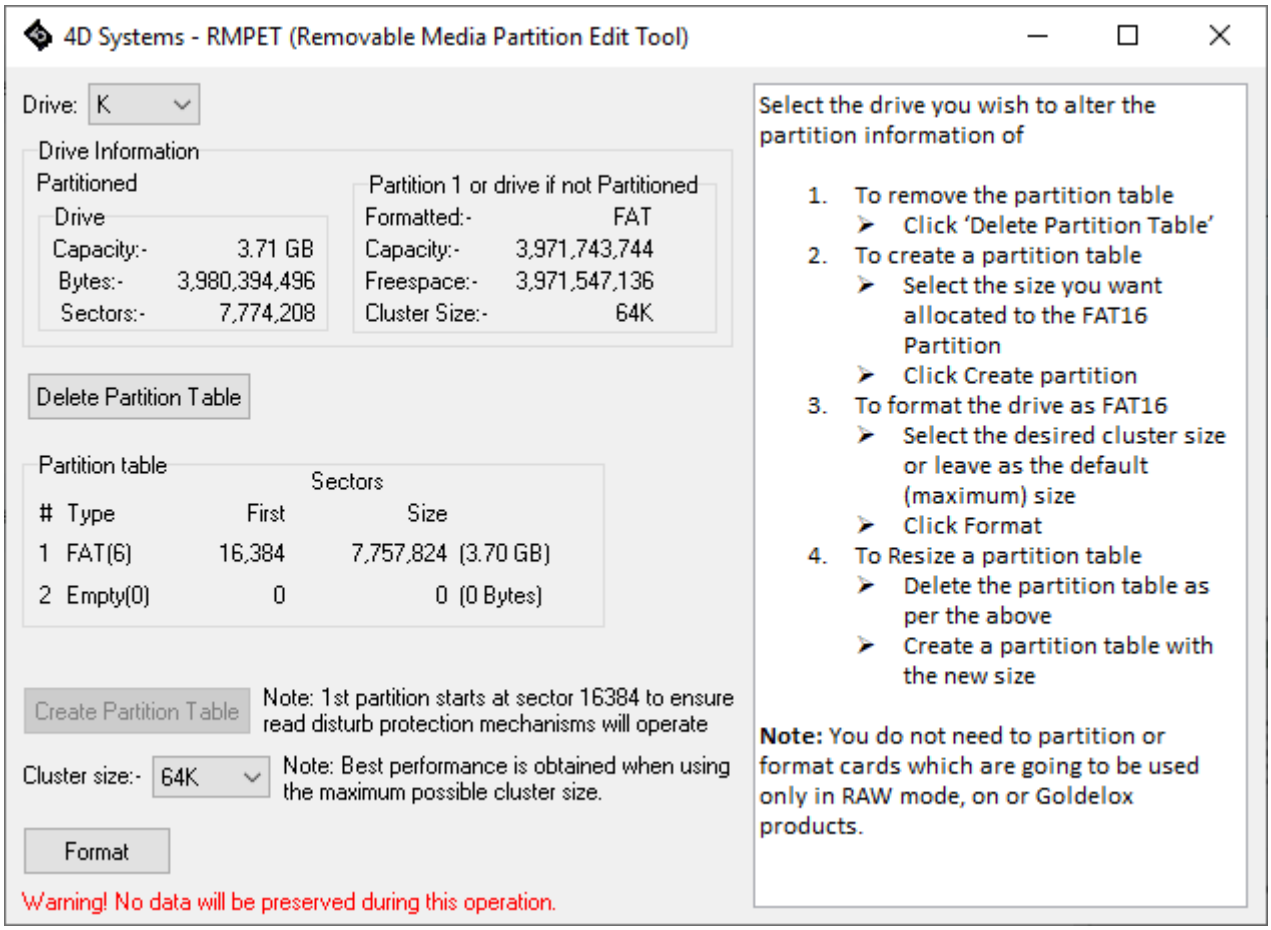


1.4. MicroSD Card Format

The microSD card shall be FAT16-formatted for processors other than GOLDELOX which uses RAW mode. It is recommended to use Workshop4's built-in utility, RMPet when formatting and partitioning a microSD card. Particularly when using uSD cards with capacity greater than 4GB.



The software provides detailed information on the uSD card's current status an easy way to partition the uSD card while considering the read disturb protection of uSD cards.



Note

Cards **MUST** be formatted FAT16 (except GOLDELOX) or RAW to work with 4D Systems models. FAT32, exFAT, NTFS etc will not work. For best results, format and partition your microSD cards using RMPet.

For a more detailed instruction on how to use this utility, please refer to 4D Systems' application note titled:

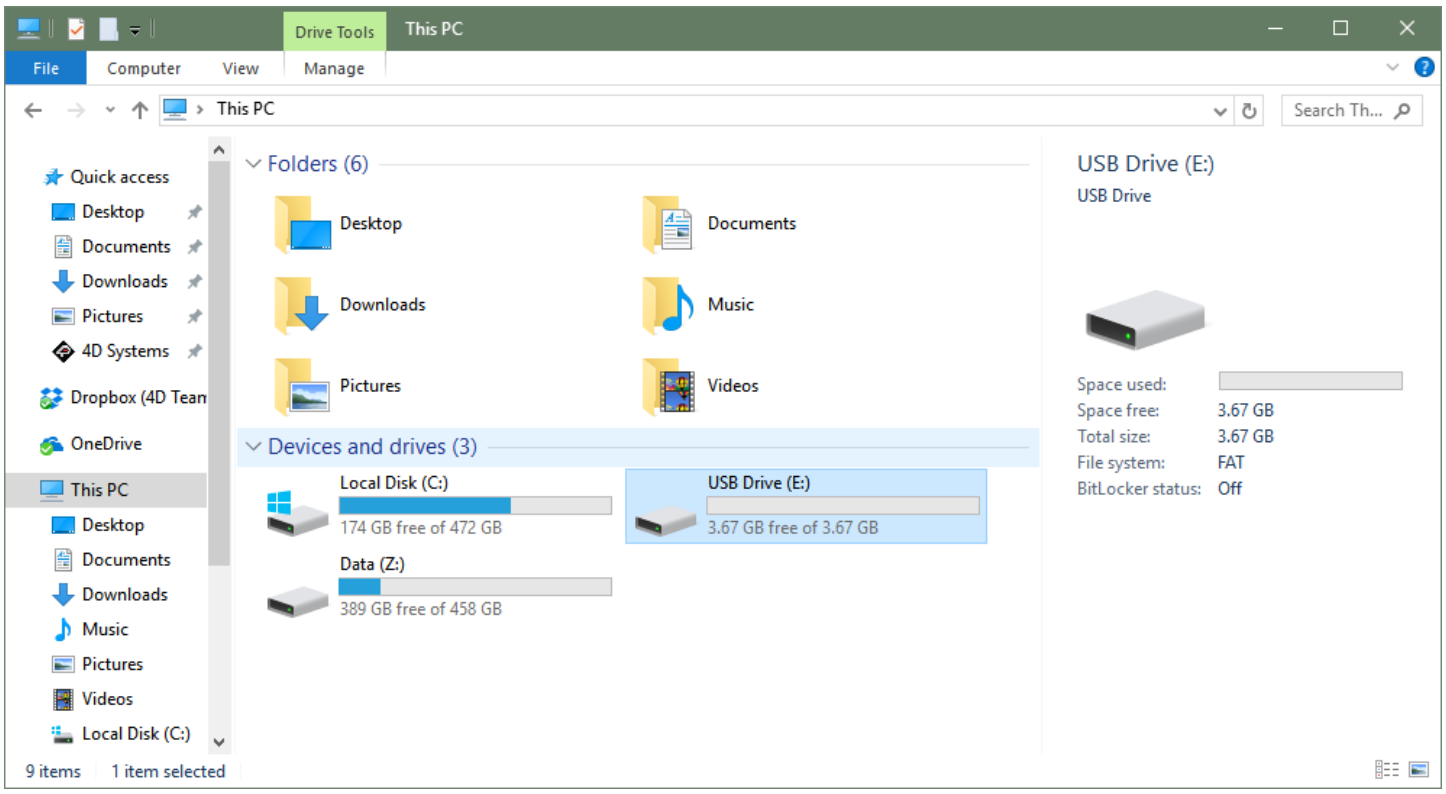
General Partitioning a microSD into FAT and RAW Components

When using RMPet to format and/or Partition your microSD card, it is best to use a microSD to USB adaptor, or a microSD to SD adaptor into a media slot of your Laptop/PC. The cards need to be formatted on your PC, not on the display module themselves. Many types/brands are available, choose one that best suits your hardware setup.

Note

microSD cards **MUST** be SPI compatible, and it is highly recommended to use Industrial Grade cards to prevent corruption over time due to a phenomenon called Read Disturb, which affects NAND Flash memory. 4D Systems offers such cards, available on our website.

Afterwards, check if the uSD card mounted successfully. Here, it is shown as drive **E:**.



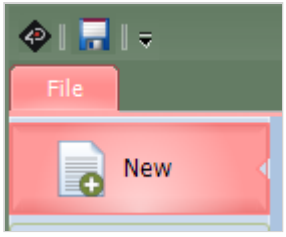
2. Creating a New Project

Launch Workshop4 just like any typical Windows application.

At launch, Workshop4 will display the **Recent** page:

From here, you have multiple options to create a new project:

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



- Click on the **Create a new 4D Systems Project** button.

- Click on the **Create a new Project** button to create a project instance based on your last project settings.

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

You can follow the image above to do the following:

1. Filter the list of display by selecting a display category
2. Select the display from the list
3. Select the desired display orientation
4. Confirm the selection

After selecting the display, the environment needs to be selected. Depending on your display you may be prompted to select either:

- a 4D Environment (when the display primarily uses a 4D processor) or;
- an Arduino Compatible Environment (when the 4D display is to be used with an Arduino compatible board or when using 4Duino)

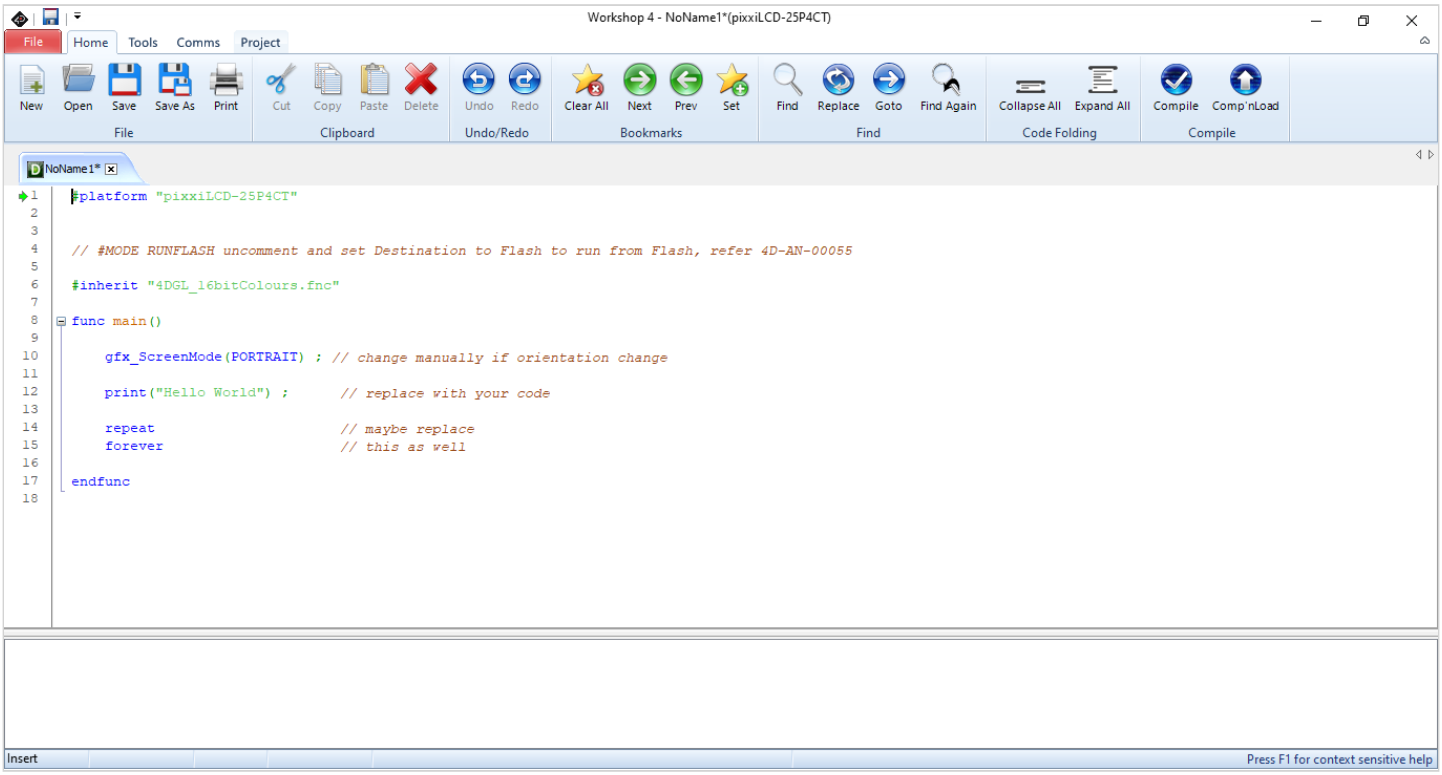
> **Note**
>
> IoD products will directly open a suitable Arduino compatible environment after product selection

3. 4D Environments

3.1. Designer

Choose the Designer environment to write 4DGL code in its raw form.

The Designer environment provides the user with a simple yet effective programming environment where pure 4DGL code can be written, compiled and downloaded to the range of 4D Systems intelligent display modules.



The designer is a very powerful environment, for those used to developing without any form of GUI aid, or for those developing complex systems where no aid is required.

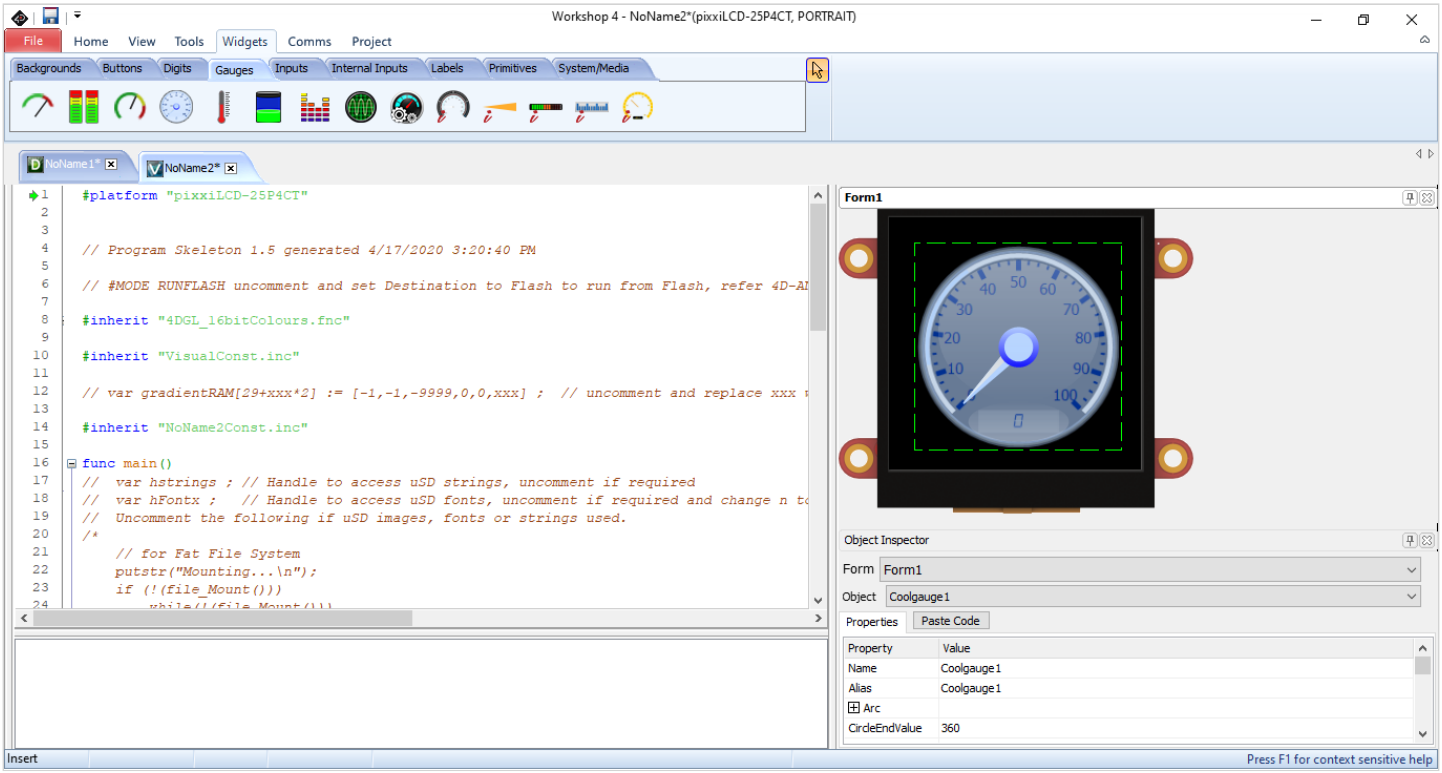
To learn more, please to refer to the Internal Functions Manual for the processor used by the display module and 4DGL Programmer's Reference Manual.

Application Notes are also available for further information.

3.2. ViSi

ViSi was designed to make the creation of graphical displays a more visual experience.

ViSi is a great 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 placed onto the simulated module display. From here each object can have its properties edited, and at the click of a button, all relevant 4DGL code associated with that object is produced in the user program. The user can then write 4DGL code around these objects to utilise them in the way they choose.

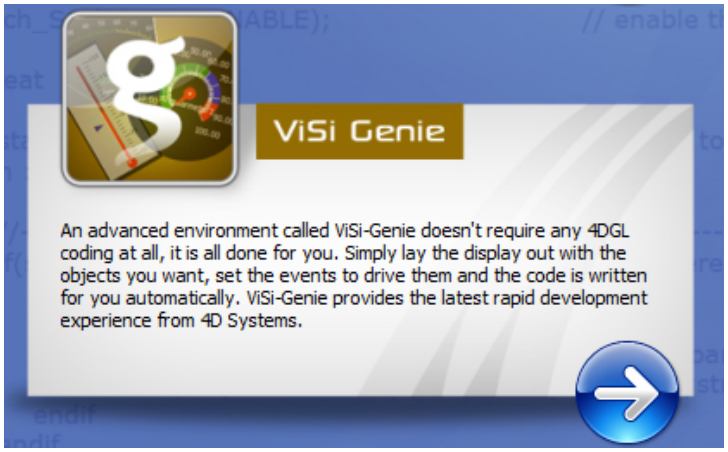


To learn more, please to refer to the ViSi User Manual, Internal Functions Manual for the processor used by the display module and 4DGL Programmer's Reference Manual.

Application Notes are also available for further information.

3.3. ViSi-Genie

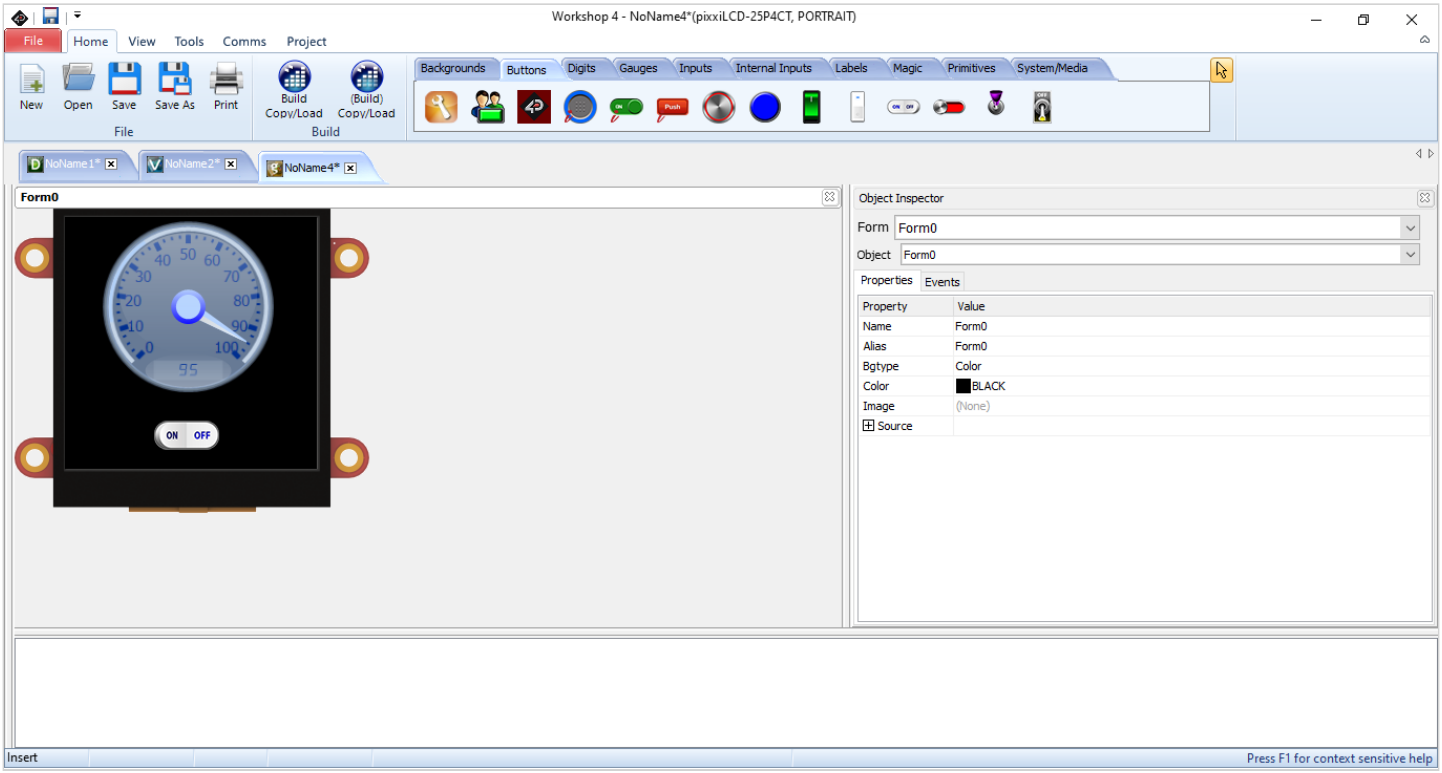
ViSi-Genie is a breakthrough in the way 4D Systems' graphic display modules are programmed. It is an environment like no other, a code-less programming environment that provides the user with a rapid visual experience, enabling a simple GUI application to be 'written' from scratch in literally seconds.



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

Pick and choose the relevant objects to place on the display, much like the ViSi Environment yet without having to write a single line of code. Each object has parameters that can be set, and configurable events to animate and drive other objects or communicate with external devices.

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



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

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

Workshop4 PRO adds a professional set of features to the Visi-Genie environment called Genie-Magic. The added features allow the user to add in 4DGL scripts, which can be activated by the display itself, from an interfacing Host, or from an external sensor or device.

These PRO set of features of Genie-Magic allow the User to create an immensely powerful GUI system with a fraction of the effort required by other systems.

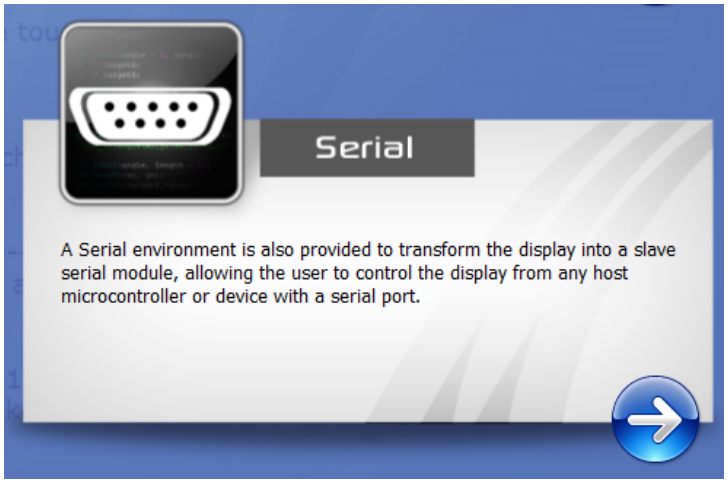
To learn more, please to refer to the ViSi-Genie User Manual.

If using ViSi-Genie Magic, information about writing code in 4DGL can be found in the Internal Functions Manual for the processor used by the display module and 4DGL Programmer's Reference Manual.

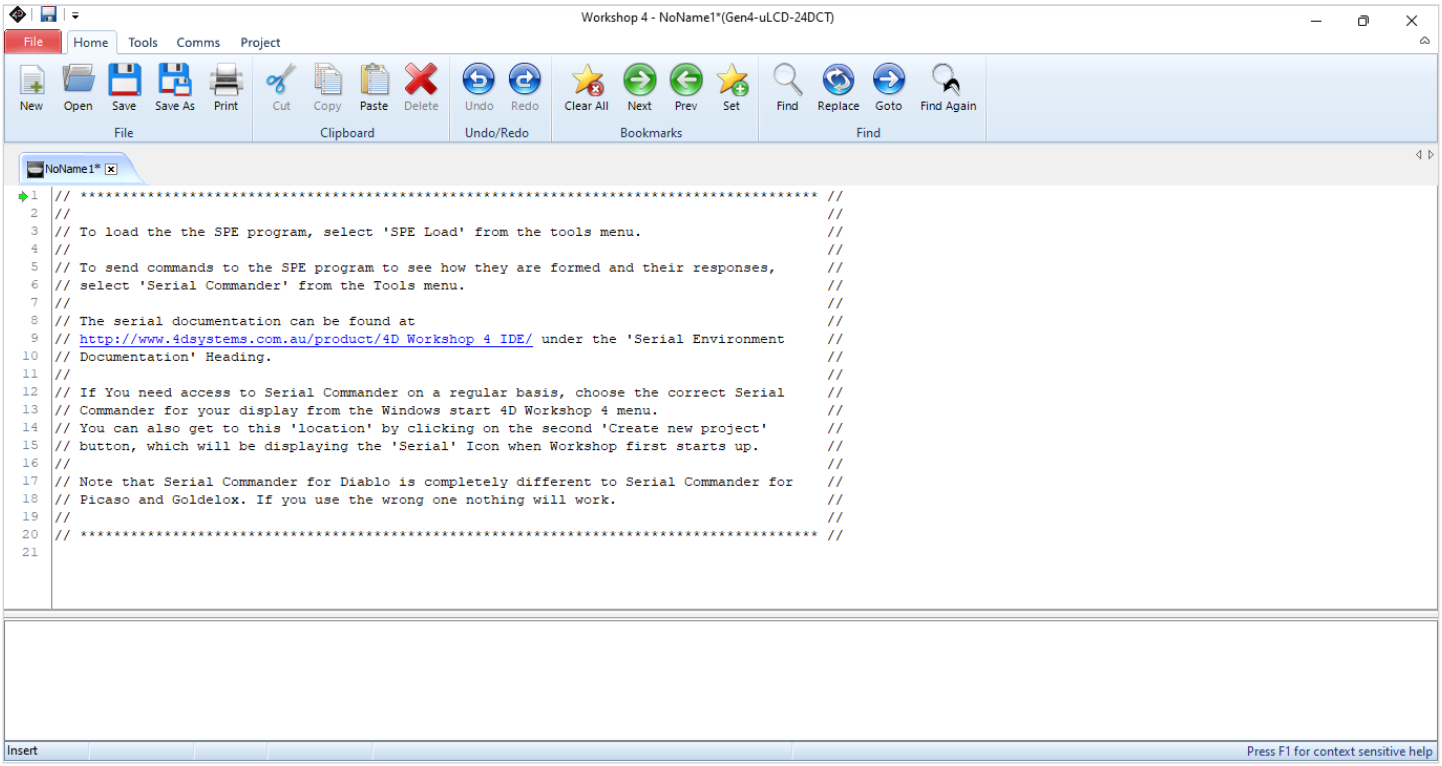
Application Notes are also available for further information.

3.4. Serial

The Serial environment (SPE) in the Workshop 4 IDE provides the user the ability to transform a 4D Systems Intelligent Display, into a slave serial graphics controller.



This enables the user to use their favourite microcontroller or serial device as the Host, without having to learn 4DGL or program in a separate IDE.



Once the display module is configured by the Serial Environment (by downloading what is called the SPE application to the module), commands can be sent from the user's host microcontroller to display primitives, images, sound and video, and can even be used to display ViSi generated graphics and widgets.

The Serial Environment should not be taken as being basic in terms of its capabilities, as it has the full 4DGL command set behind it, but available from the Host rather than from programming the display module itself using the Workshop4 IDE.

Virtually anything created in Designer or ViSi can be designed or controlled from the Serial Environment.

For ease of development for an Arduino compatible modules as host in Serial (SPE), consider using the Arduino Compatible Environments

Please refer to the following reference manuals for a complete listing of all the supported Serial commands for each processor

- DIABLO-16 Serial Command Set Reference Manual
- PIXXI Serial Command Set Reference Manual
- PICASO Serial Command Set Reference Manual
- GOLDELOX Serial Command Set Reference Manual

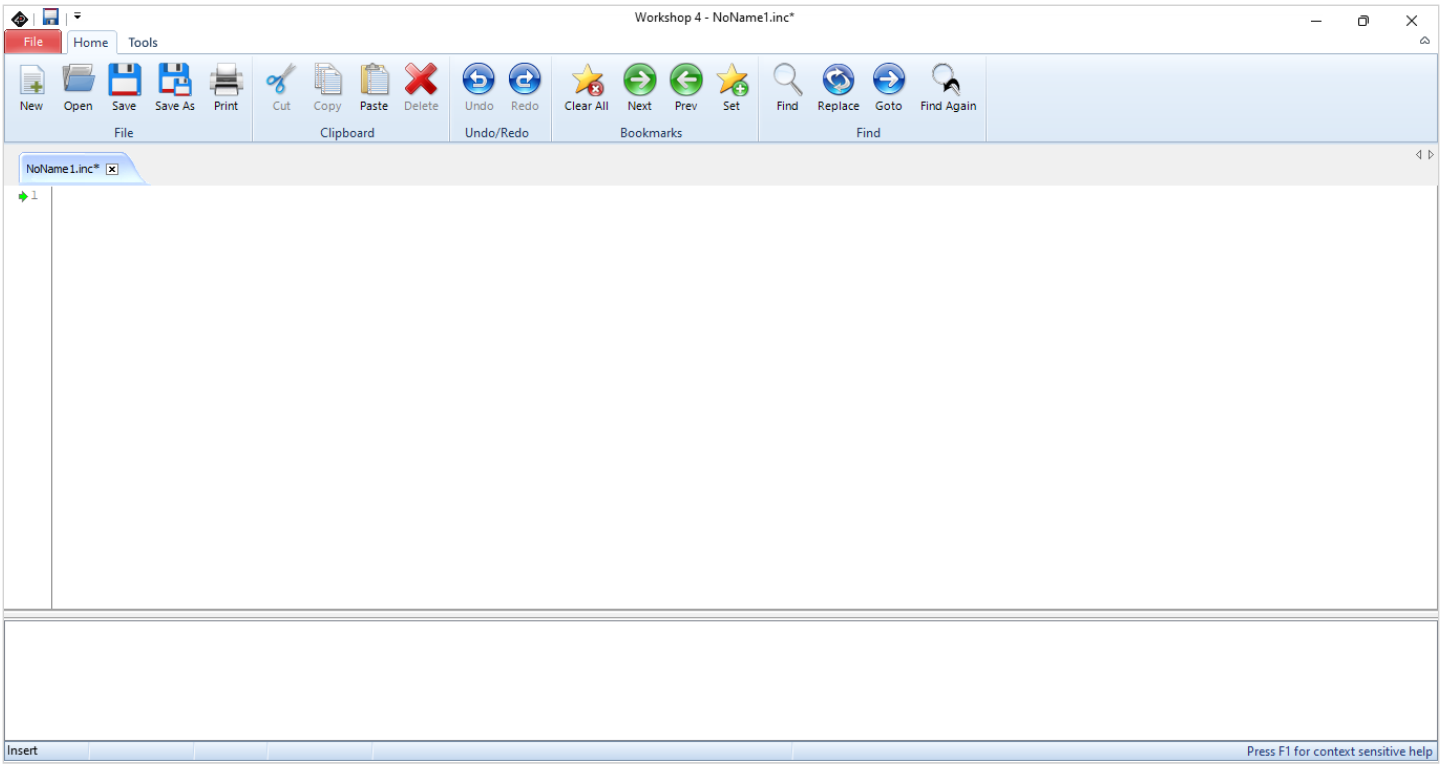
3.5. Additional Options

Additional Editor options are provided instead of the 4 environments.

This serves to provide a way to create a system or text file that may be used together with the basic environments.

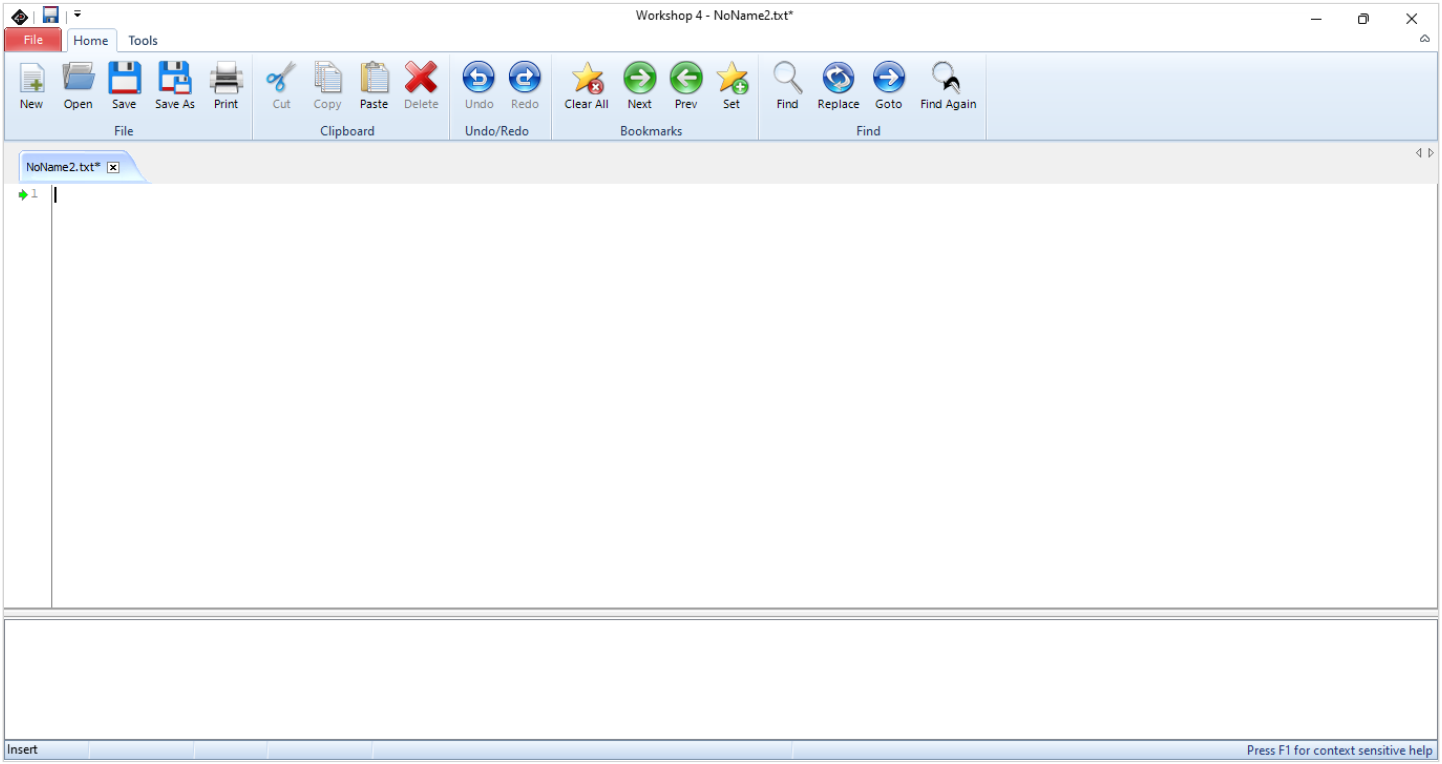
3.5.1. Create System File

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.



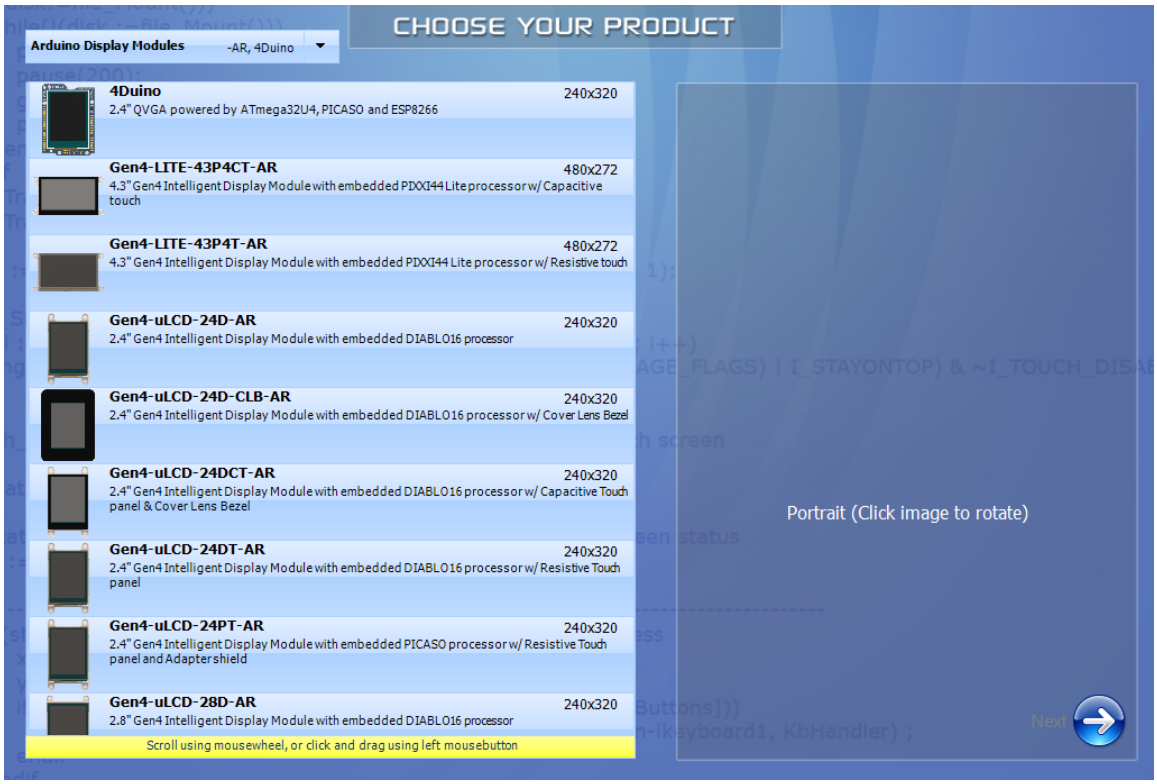
3.5.2. Create Text File

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



4. Arduino Compatible Environments

When choosing a 4D Systems product, you can see that there is a product group named **Arduino Display Modules** which includes both the 4Duino and a wide range of 4D Systems' display modules setup with an Arduino board (**-AR** modules).



If a product under this category is selected, a different set of environments is provided allowing the users to write their Arduino code and customize their 4D graphics user interface at the same time in a single integrated development environment.

4.1. 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.

4.2. 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 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.

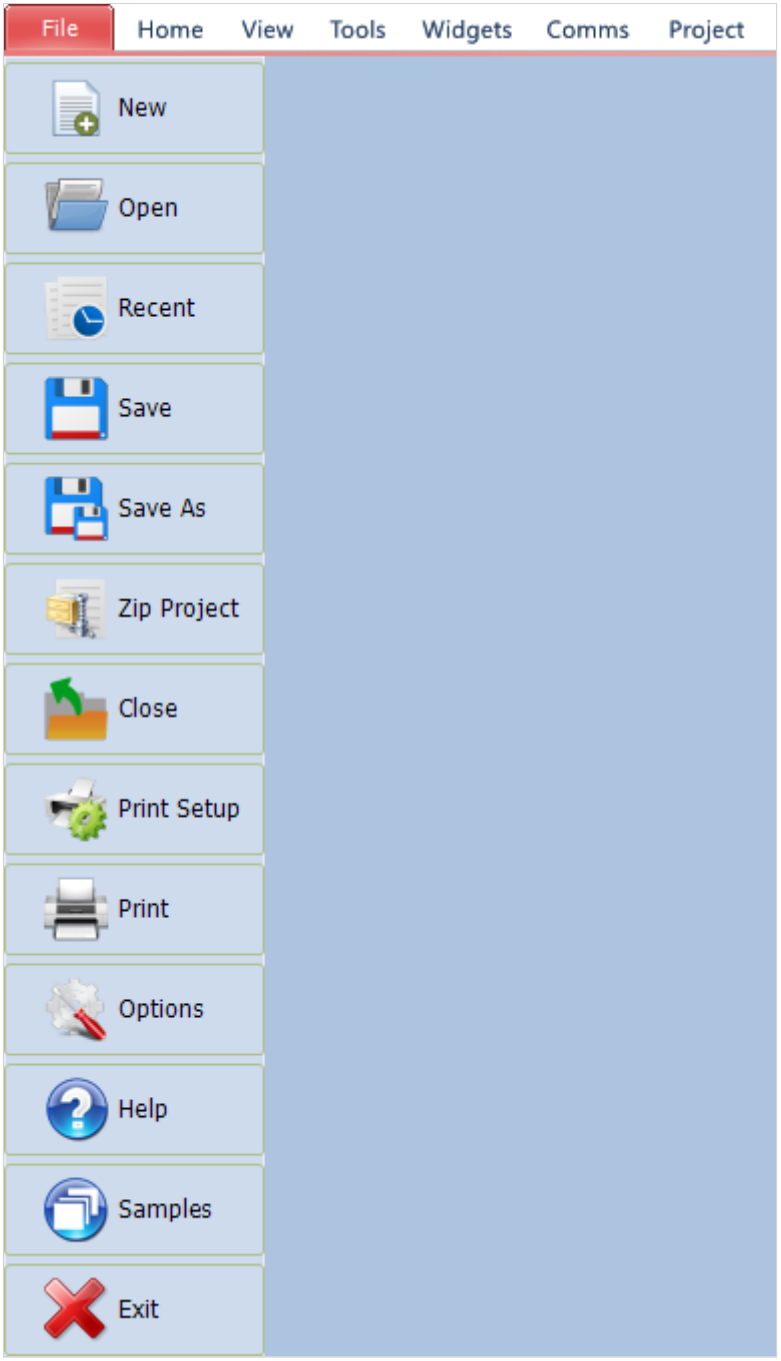
4.3. Genie Graphics

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.

5. Common File Menu

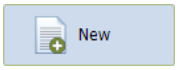
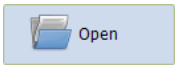
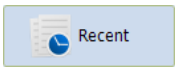
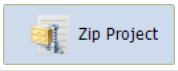
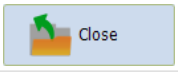
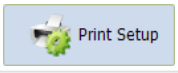
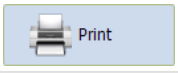
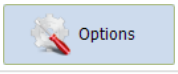
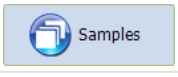
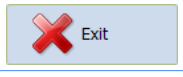
The **File** Menu is the first menu and common to all environments.



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

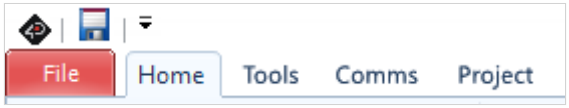
- File-related buttons,
- Print-related buttons,
- And miscellaneous buttons, such as Help, Options and Samples

File Menu Buttons

Button	Description
	Create a New Project by selecting the target display module and development environment
	Open a standard Open file window to browse and load an existing project
	Displays a list of recently accessed files and options to create a new project from scratch or based on last used project setting
	Save all modified projects/files
	Save a copy of a previously saved project/file and give it a new name
	Make a compressed file out of the project. This is especially useful when sharing projects.
	Close the current project. This will open a save prompt in case there are unsaved changes.
	Open the Print Setup window that can be used to setup the printer and print properties
	Open a Print window that can be used to print the project
	Provide an interface to modify the IDE settings and the default options for a **NEWLY** created projects
	Provide links to website, community forum, and other helpful web content
	Provide a list of basic example project from 4D Systems
	Close Workshop4. This will open a save prompt in case there are unsaved changes.

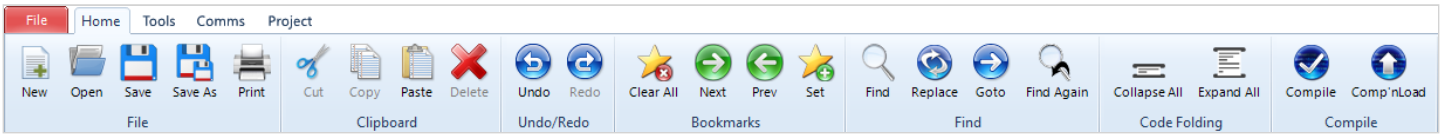
6. Designer Specific Menus

The Designer environment includes five menus:



6.1. Home Menu

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



This ribbon menu contains the following button groups

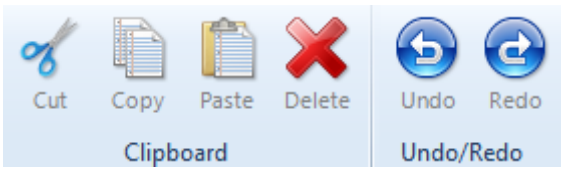
- File-related buttons
- Code-related buttons
- Bookmark buttons
- Find and Replace Buttons
- Compile Buttons

6.1.1. File-Related Buttons

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

6.1.2. Code-Related Buttons

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



6.1.3. Bookmark Buttons

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

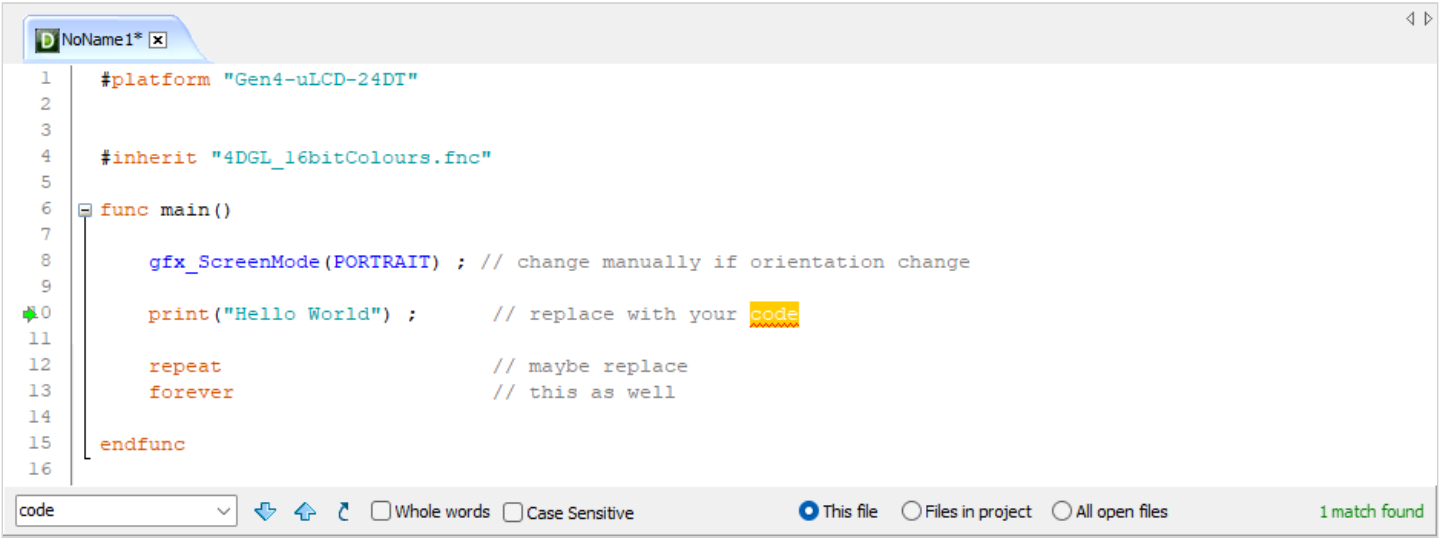
Bookmarks are shown close to the line number:

Bookmarks are especially useful for large projects.

6.1.4. Find and Replace Buttons

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

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



Use the up and down arrows to look for the previous and next occurrence.

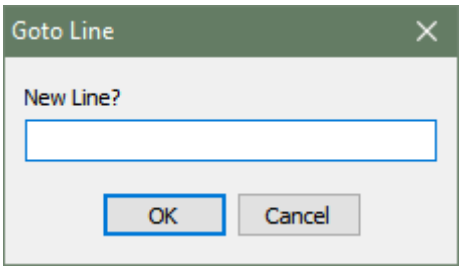
Check **Whole Words** and **Case Sensitive**.

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:

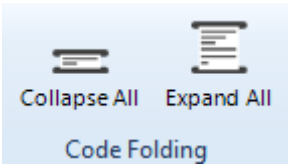
Same options as for **Find** apply to this with the addition of **Replace** and **Replace all** buttons.

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



6.1.5. Code Folding Buttons

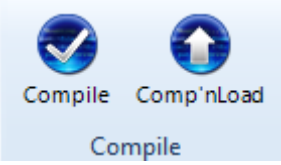
The code folding buttons allow to collapse or expand a function:



This is especially useful for large projects.

6.1.6. Compile Buttons

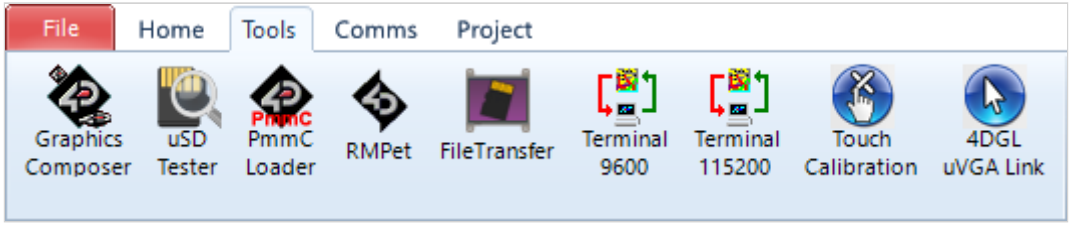
The **Compile** button launches the compilation of the project, can also be accessed with shortcut key Ctrl + F9. The **Comp'nLoad** compiles and uploads the project to the screen, which has a shortcut key of F9.



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

6.2. Tools Menu

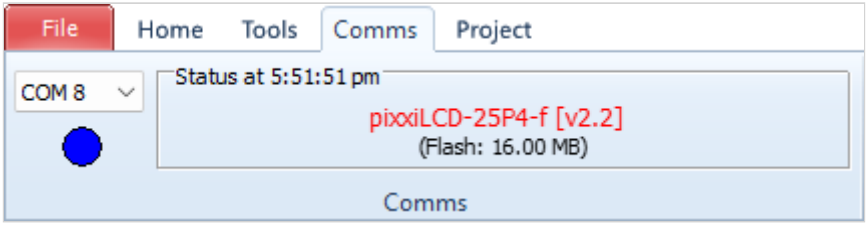
All the tools and utilities useful for the Designer environment are included here.



Tools Menu Buttons	
Graphics Composer	Open the Graphics Composer tool. This legacy 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.
uSD Tester	Open the uSD Tester tool 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.
PmmC Loader	Open the PmmC Loader tool to update the PmmC/Driver for the target display
RMPet	Open the RMPet tool to partition the uSD card. The uSD card must be mounted to the PC.
FileTransfer	Open the File Transter utility allowing read and write operations to a display modules' uSD or external flash via USB-to-Serial connection
Terminal 9600	Open the Terminal tool and connect to the currently selected port at 9600 baud
Terminal 115200	Open the Terminal tool and connect to the currently selected port at 115200 baud
Touch Calibration	Load a touch calibration program to *resistive* display modules connected at selected port
4DGL uVGA Link	Open an interactive window to use mouse/keyboard with the uVGA-II or -III module

6.3. Comms Menu

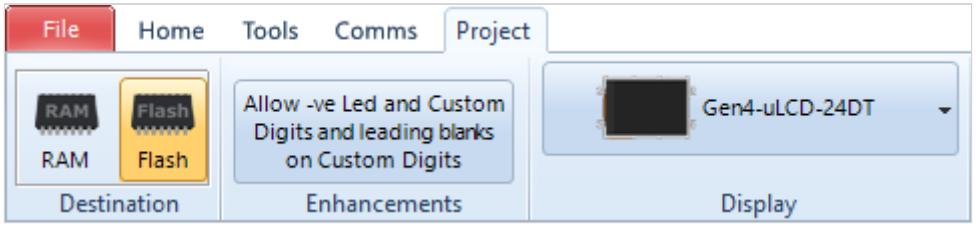
This menu oversees the communication port:



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

6.4. Project Menu

The Project menu includes different parameters and options

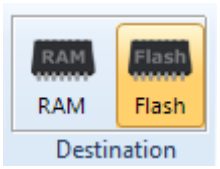


There are three areas in this menu.

- Destination,
- Enhancements,
- And display selection.

6.4.1. Destination

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

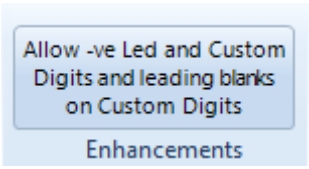


Select the **Destination** among two options:

- **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 selected as the destination, the program is lost when the display is turned off. This is for testing during development, 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 selected as the destination, the program is retained and will be available after power cycling. This should be the default for normal use.

6.4.2. Enhancements

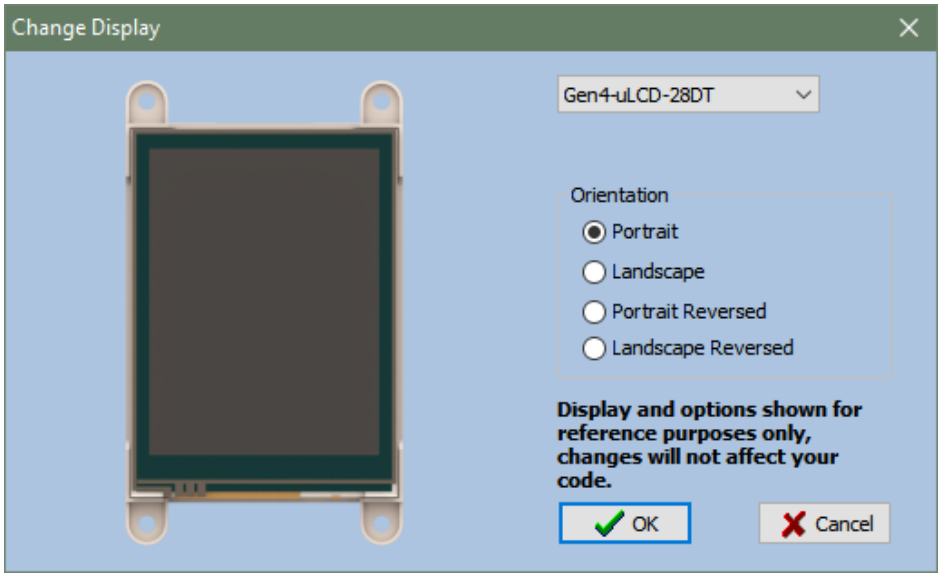
The second section contains a button for enabling the use of negative values for LED digits and custom digits objects and for enabling the use of leading blanks on custom digit objects.



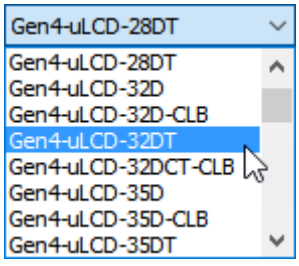
6.4.3. Display Selection

The third section allows to select the screen.

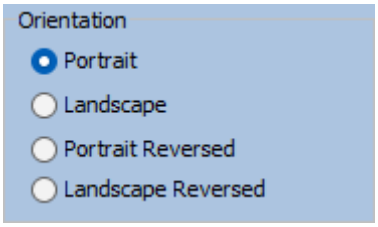
Clicking on the button opens a new window to select the screen, and adjust the orientation of the module to suit how the module will be mounted:

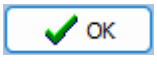
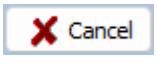


Select the screen from the drop-down list:



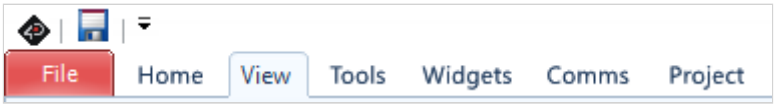
Define the orientation among the four options:



Confirm by clicking on  or deny by clicking on 

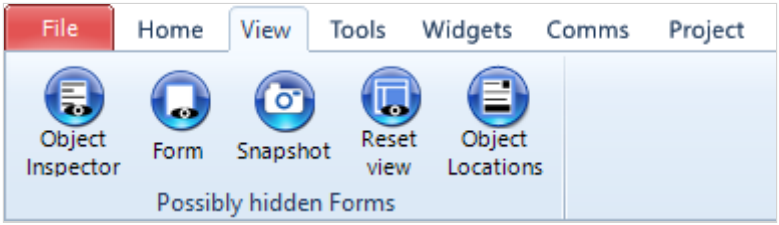
7. ViSi Specific Menus

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.

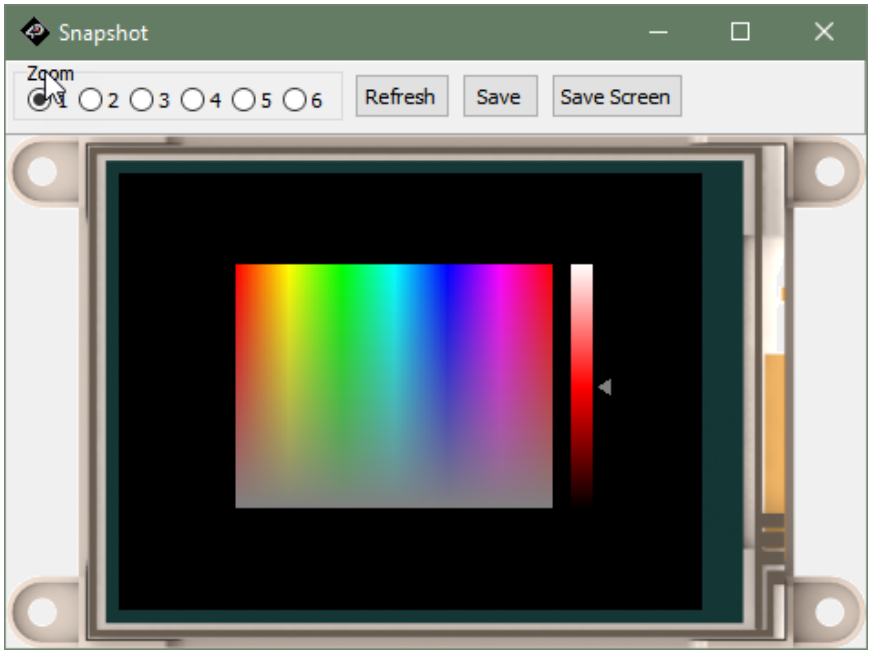


7.1. View Menu

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 be made.



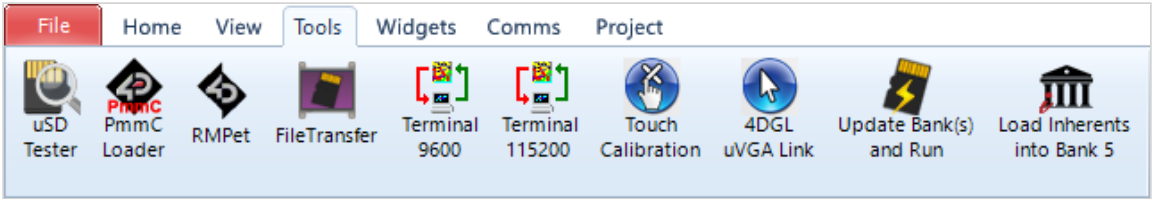
This window provides a zoom up to 6 times. The **Save** button allows to save the screen as an image.

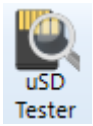
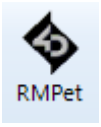
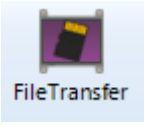
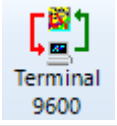
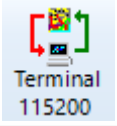
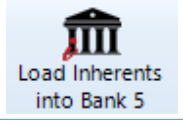
Object Inspector, Form, and Reset View, all relate to the menus and tool bars used in Workshop4. These toolbars and menus can be moved and detached from the side of Workshop4. Object Inspector and Form will bring to front the relevant toolbar when clicked. If required, the toolbars can be reset back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

7.2. Tools Menu

All the tools and utilities useful for the ViSi environment are included here.



Tools Menu Buttons	
	Open the uSD Tester tool 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.
	Open the PmmC Loader tool to update the PmmC/Driver for the target display
	Open the RMPet tool to partition the uSD card. The uSD card must be mounted to the PC.
	Open the File Transter utility allowing read and write operations to a display modules' uSD or external flash via USB-to-Serial connection
	Open the Terminal tool and connect to the currently selected port at 9600 baud
	Open the Terminal tool and connect to the currently selected port at 115200 baud
	Load a touch calibration program to *resistive* display modules connected at selected port
	Open an interactive window to use mouse/keyboard with the uVGA-II or -III module
	See Update Bank(s) and Run (*only for DIABLO-16 displays*)
	See Load Inherents into Bank 5 (*only for DIABLO-16 displays*)

Update Bank(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.

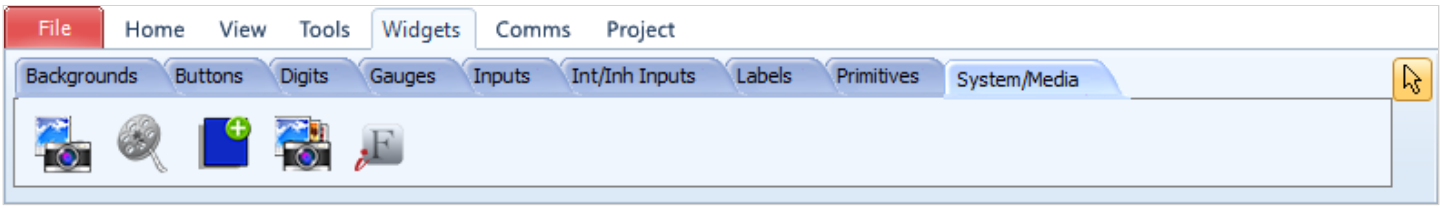
It is also possible to modify the **Update Bank(s) and Run** program such that it only updates the target banks only if the corresponding ViSi program files in the uSD card have a newer time and date information.

Load Inherents into Bank 5

This is a tool which loads the widget software related to **Inherent Widgets** when used on a DIABLO-16 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.

7.3. Widgets Menu

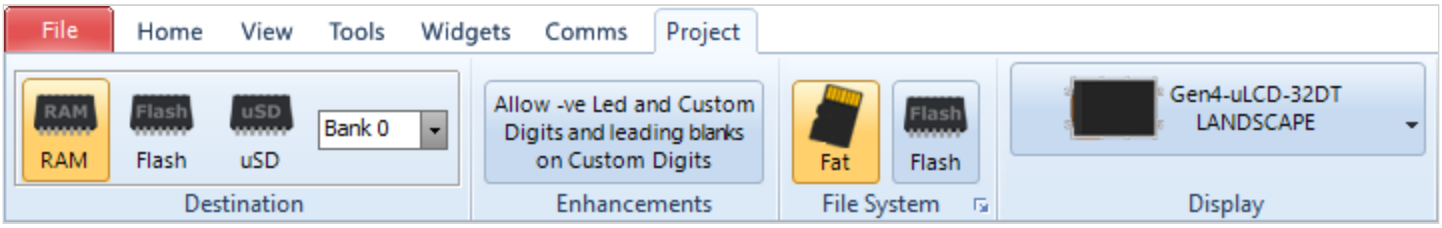
The **Widgets** menu includes the objects pane with all the objects available to build the interface:



See Workshop4 Widgets for more information.

7.4. 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.

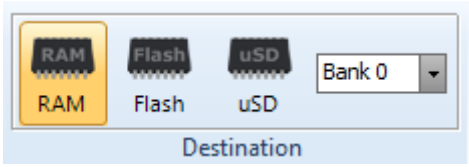


There are four areas in this menu.

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

7.4.1. Destination

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

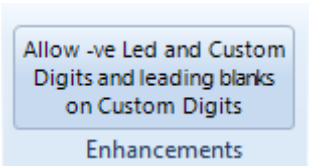


Select the **Destination**:

- **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 **Tools** menu. 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 DIABLO-16.

7.4.2. Enhancements

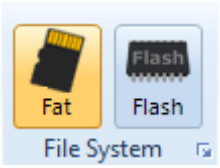
The second section's button allows for enabling the use of negative values for LED digits and custom digits objects and for enabling the use of leading blanks on custom digit objects.



This is selectable as some older projects were made before this option was enabled, so it is required to be user selected to retain historical functionality.

7.4.3. 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:

These 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.

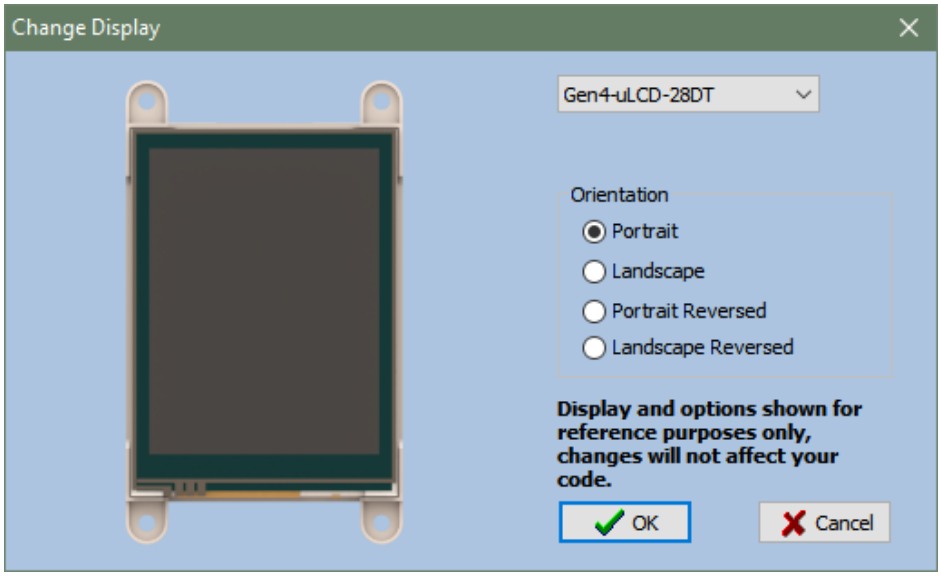
> **Note**
>
> GCI widgets can become large, and External Flash is limited to typically 16MB for 4D Display modules, so there is a very real chance that GCI widgets may not fit in External Flash, and other widget types such as Internal or Inherent may need to be utilised.

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, please refer to Workshop4 Widgets section.

7.4.4. Display Selection

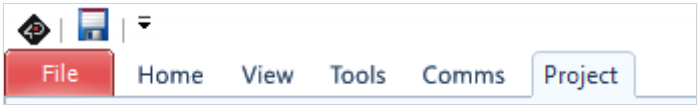
The last section allows selecting the screen, useful for converting a program from one display module to another.



Please refer to Display Selection section of Designer Specific Menus for more information.

8. ViSi-Genie Specific Menus

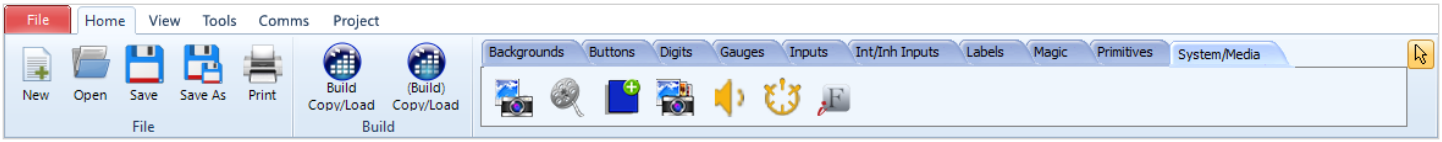
ViSi-Genie includes five menus with specific ribbons and options.



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

8.1. Home Menu

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



This ribbon menu contains the following button groups

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

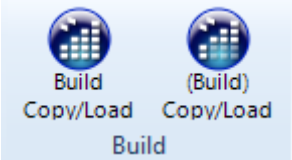
8.1.1. File-Related Buttons

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

8.1.2. Build Buttons

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`

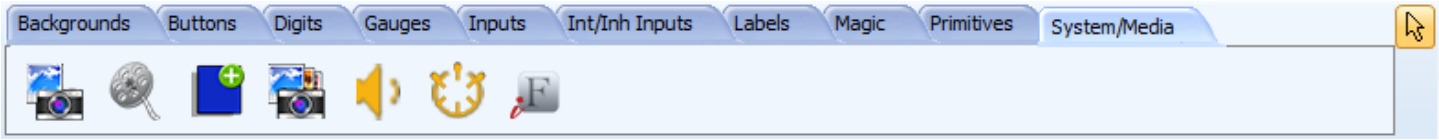
The **(Build)** button will only compile or generate the GUI if required if there have been changes, and then uploads the project to the display module, and offers to transfer media to uSD/Flash storage. Shortcut Key is `F9`



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.

8.1.3. Objects Pane

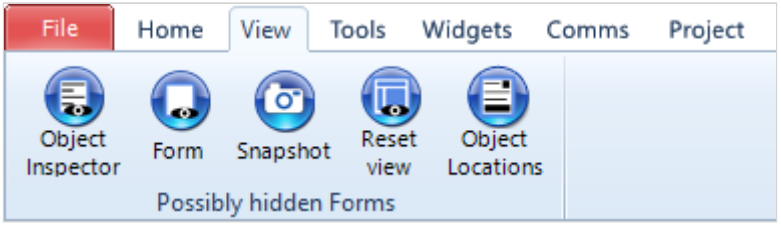
The objects pane includes all the objects available to build the interface:



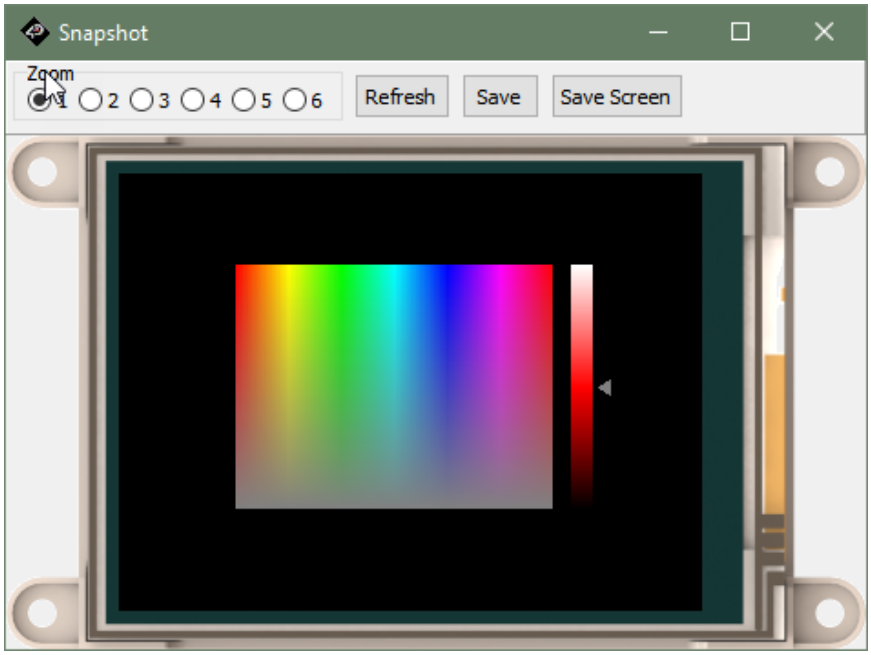
Please refer to the ViSi-Genie Manual for more information.

8.2. View Menu

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



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



This window provides a zoom up to 6 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.

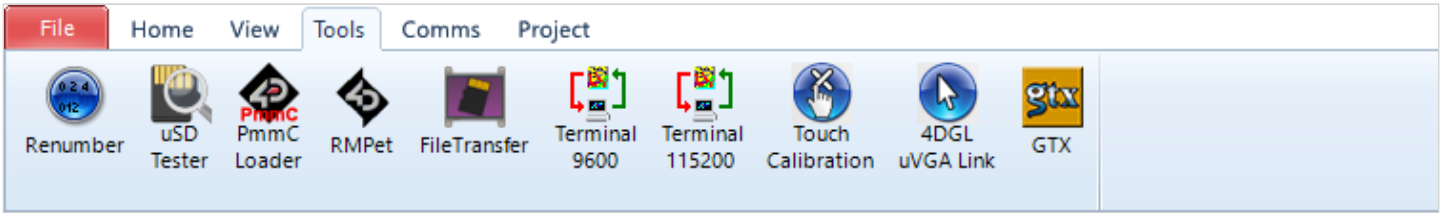
Object Locations enables the user to copy the locations/coordinates of objects on the display, to the clipboard.

8.3. Tools Menu

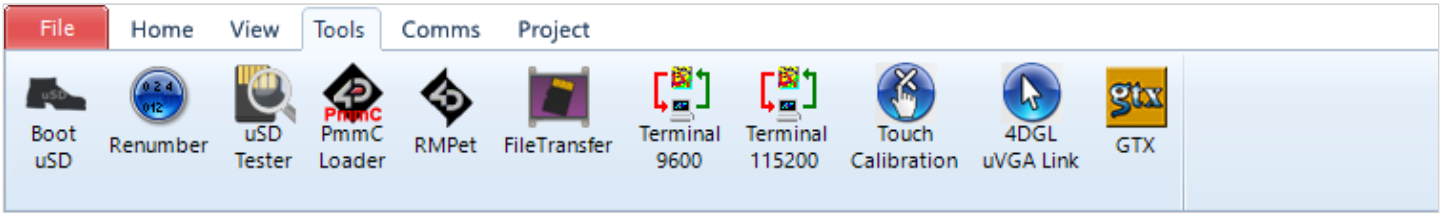
All the tools and utilities useful for the Genie environment are included here.

The tools available depends on the processor of the target display module:

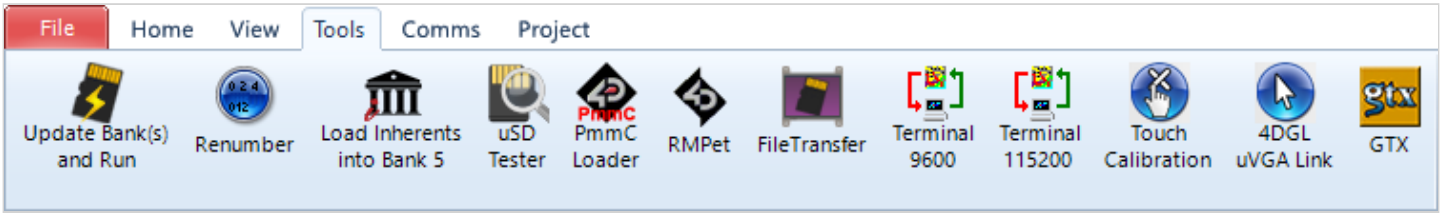
PIXXI-



PICASO

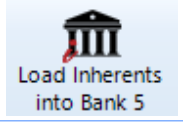


DIABLO-16



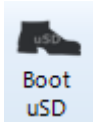
For **DIABLO-16** display modules, the **Update Bank(s) and Run** button and **Load Inherents into Bank 5** tools are added. For more information regarding these tools, please check the application notes.

DIABLO16-only Tools

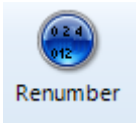
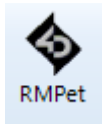
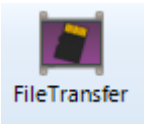
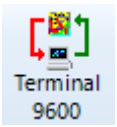
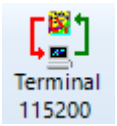
	See Update Bank(s) and Run (*only for DIABLO-16 displays*)
	See Load Inherents into Bank 5 (*only for DIABLO-16 displays*)

For **PICASO** display modules, the **Boot uSD** button is added. For more information regarding this tool, please check the application notes.

PICASO-only Tools

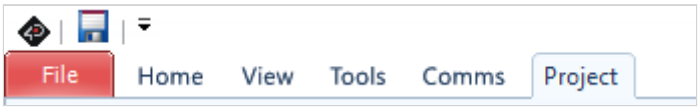
	Upload the Boot uSD application to the screen, enabling programs to be loaded via microSD card (*only for PICASO displays*)

Common Tools

	Reallocate the indexes of all the widgets in your application, as during development widgets may be deleted, leaving gaps in the numbering index.
	Open the uSD Tester tool 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.
	Open the PmmC Loader tool to update the PmmC/Driver for the target display
	Open the RMPet tool to partition the uSD card. The uSD card must be mounted to the PC.
	Open the File Transter utility allowing read and write operations to a display modules' uSD or external flash via USB-to-Serial connection
	Open the Terminal tool and connect to the currently selected port at 9600 baud
	Open the Terminal tool and connect to the currently selected port at 115200 baud
	Load a touch calibration program to *resistive* display modules connected at selected port
	Open an interactive window to use mouse/keyboard with the uVGA-II or -III module
	Open the Genie Test eXecutor application providing a host simulator that can display commands being sent and received

8.4. Project Menu

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

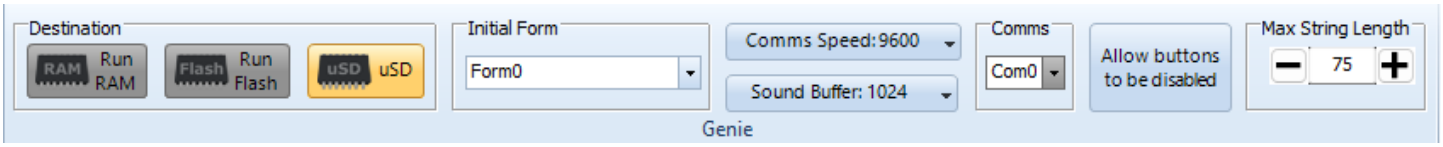


There are four groups of buttons:

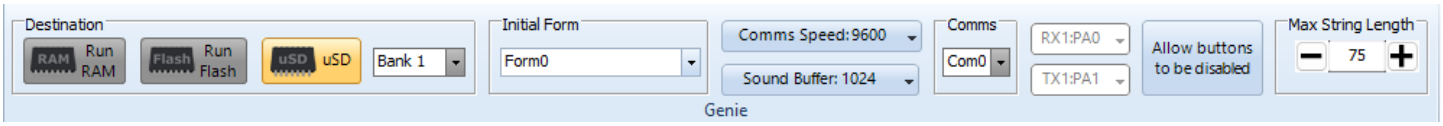
- Options for Genie,
- Enhancements,
- File System,
- And display selection.

8.4.1. Genie Options

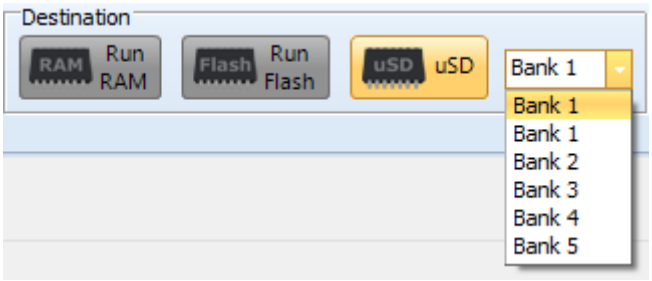
PICASO and PIXXI share the same Genie options:

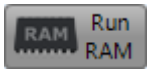
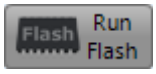
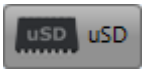
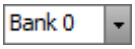


DIABLO on the other hand has an additional option for Program **Destination**.



This provide a selection for DIABLO-16's flashbanks.



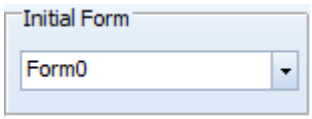
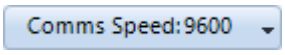
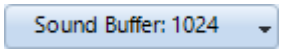
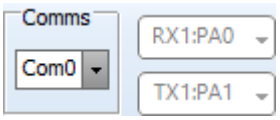
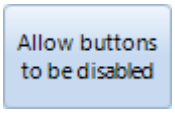
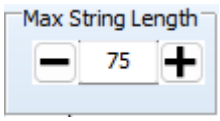
Program Destination	
	The display must be connected during Build and the program code will be downloaded to the display's flash memory (Processors Flash, not external flash) once compiled. The user's application will be stored in Flash but will be run from RAM.
	The display must be connected during Build 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.
	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.
	The additional drop-down menu allows the user to specify the target destination flash bank of the ViSi-Genie program. The DIABLO-16 processor has six flash banks (Bank 0 to Bank 5), each of which has a capacity of 32 kB.

When using **DIABLO-16** Flashbanks:

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.

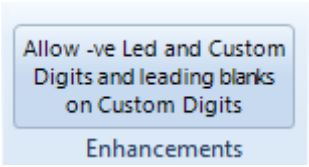
On the other hand, when **uSD** is selected, the ViSi-Genie program will be copied to the uSD card. This option does not require the display module to be connected to the PC during build time. However, this option requires the **Update Bank(s) and Run** program to be downloaded to Bank 0 of the display's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

Additional Genie Options

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

8.4.2. Enhancements

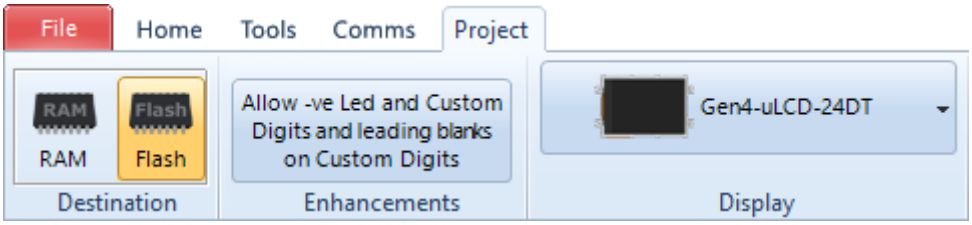
This button allows for enabling the use of negative values for LED digits and custom digits objects and for enabling the use of leading blanks on custom digit objects.



This is selectable as some older projects were made before this option was enabled, so it is required to be user selected 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.

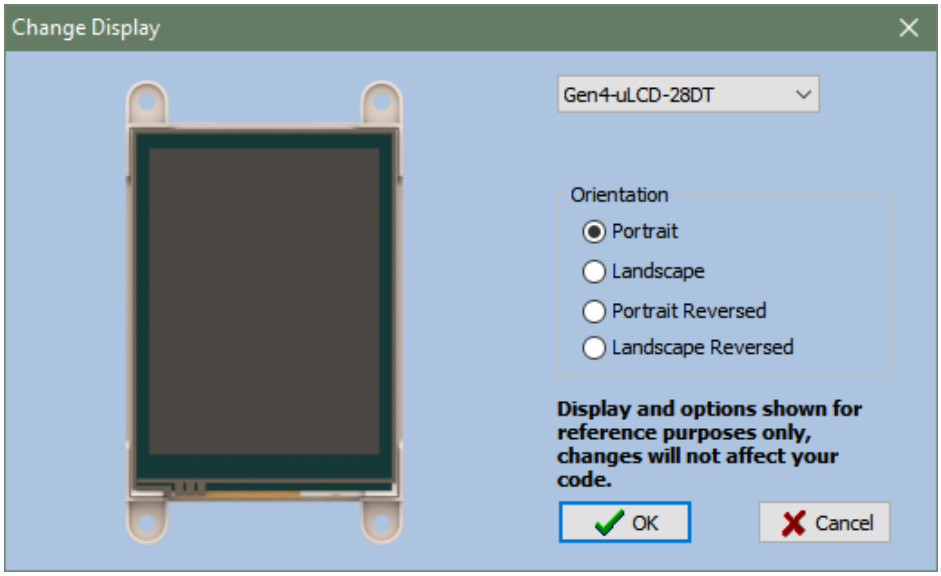


Selecting Fat will target the multimedia to be stored on the microSD card. Selecting Flash will target the multimedia to be stored on Flash Memory.

> **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. Please see File System section of ViSi Specific Menus for a short discussion of different type of widgets and the storage requirements.

8.4.4. Display Selection

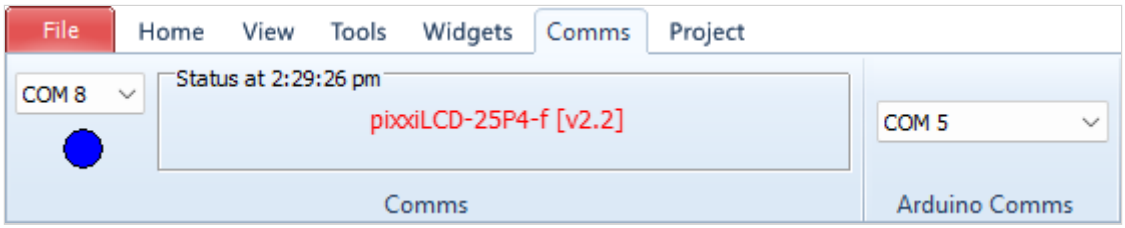
The last section allows selecting the screen, useful for converting a program from one display module to another.



Please refer to Display Selection section of Designer Specific Menus for more information.

9. Basic/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.



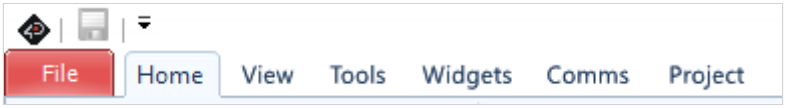
The Arduino Comms refers to the COM port that the Arduino board is currently using.

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

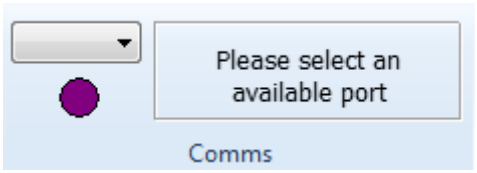
Since the Basic Environment is designed to allow users to create projects without the need for a uSD card, it only allows the user to use primitive shapes and objects in the WYSIWYG window. The Extended Graphics on the other hand gives additional support for 4D Graphics.

10. Connect the Module

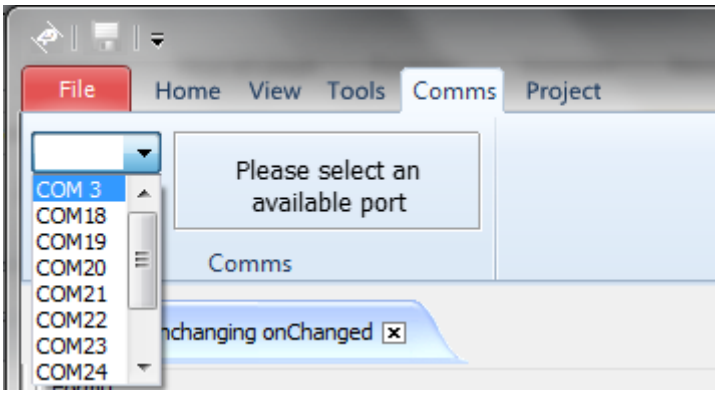
Connect the module to a USB port with the 4D Systems programming cable and select the **Comms** menu:



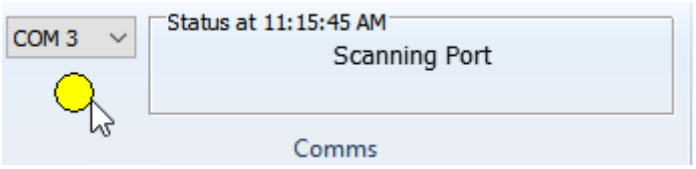
Above the Comms section, the **violet** light mentions no module is currently connected.



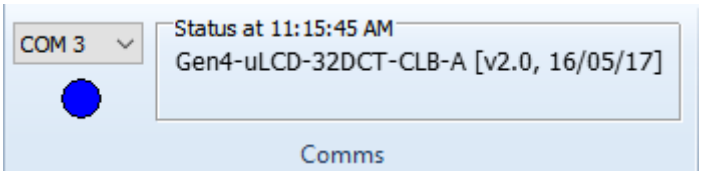
Connect the 4D Systems programming cable/adaptor to the module and plug the cable into the USB port. Click on the drop-down list and select the COM port relating to the 4D Programming cable/adaptor.



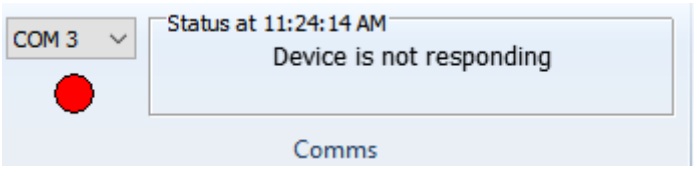
The light turns **yellow** while the connection is being established:



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



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



11. Workshop4 Widgets

Workshop4 provides a drag n' drop design tool for most of its powerful development environments. This allows user to easily utilise three types of widgets in their projects:

- GCI Widgets
- Internal Functions (PmmC) Widgets
- Inherent Widgets

This offers increased flexibility in creating applications depending on the hardware requirement. These widgets include the following: buttons, sliders, knobs, gauges, LED indicator, LED digit, strings, static labels, media, and images. Additional non-GUI widgets are also available that allows the user to add miscellaneous functionalities like sound, hardware I/O, or supporting resources like fonts, depending on the target processor and environment used. All these widgets are available in the Widgets Menu of the ViSi and ViSi-Genie environment of Workshop4 IDE. Given its high configurability, these widgets can be customized directly on the What You See Is What You Get (WYSIWYG) screen or through the **Object Inspector**.

You can refer to the Workshop4 Widgets Reference Manual for more information.

11.1. Smart Widgets

Workshop4 PRO also adds the ability to use SMART widgets, which are tools to assist the User to create their Buttons, Sliders, Gauges etc.

You can refer to the Smart Widgets User Guide for more information.

12. Revision History

Document Revision

Revision Number	Date	Content
1.0	19/11/2012	First Release
1.1	17/12/2012	Typos on Page 4 fixed
1.2	04/02/2013	Added new content for Serial and fixed incorrect document references
1.3	05/07/2013	Amended details about Micro-SD card
1.4	11/03/2014	Amended details about Program Destination
2.0	01/05/2017	Updated formatting and contents
2.1	29/07/2017	Added information on target flash banks
2.2	05/04/2019	Updated Formatting
2.3	27/07/2020	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-.
2.4	07/02/2023	Modified for web-based documentation
2.5	20/03/2024	Updated formatting for resource centre redesign

13. Legal Notice

13.1. 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 makes changes to Specifications or written material without prior notice at any time.

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

13.2. Disclaimer of Warranties & Limitations of Liabilities

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.