4D SYSTEMS
TURNING TECHNOLOGY INTO ART

ViSi Genie Magic File Size
Request Arduino

DOCUMENT DATE: **22nd April 2019**
DOCUMENT REVISION: **1.1**

Description

This application note shows how to write a sketch for an Arduino program that sends **WRITE_MAGIC_BYTES** messages to and handles **REPORT_MAGIC_EVENT_BYTES** messages coming from the display module.
Note: Workshop Pro is needed for this application.

Before getting started, the following are required:

- Any of the following 4D Picaso display modules:

uLCD-24PTU	uLCD-28PTU	uVGA-III
gen4-uLCD-24PT	gen4-uLCD-28PT	gen4-uLCD-32PT

  and other superseded modules which support the ViSi Genie environment.

- The target module can also be a Diablo16 display

gen4-uLCD-24D Series	gen4-uLCD-28D Series	gen4-uLCD-32D Series
gen4-uLCD-35D Series	gen4-uLCD-43D Series	gen4-uLCD-50D Series
gen4-uLCD-70D Series		
uLCD-35DT	uLCD-43D Series	uLCD-70DT

  Visit www.4dsystems.com.au/products to see the latest display module products that use the Diablo16 processor.

- 4D Programming Cable / µUSB-PA5/µUSB-PA5-II for non-gen4 displays (uLCD-xxx)
- 4D Programming Cable & gen4-IB / gen4-PA / 4D-UPA, for gen-4 displays (gen4-uLCD-xxx)
- micro-SD (µSD) memory card
- Workshop 4 IDE (installed according to the installation document)
- When downloading an application note, a list of recommended application notes is shown. It is assumed that the user has read or has a working knowledge of the topics presented in these recommended application notes.

Content

Application Overview

To understand this application note more quickly, the reader is advised to read and understand first the following application notes.

ViSi-Genie Magic File Size Request

ViSi-Genie Magic How to Append to a File

ViSi-Genie Magic How to Append to a File Arduino

The application notes **ViSi-Genie Magic File Size Request** and **ViSi-Genie Magic How to Append to a File** show how the Magic Object is used to implement a ViSi-Genie project that allows the host to access files on the uSD card of the display. The Magic Object is one of the objects under the Magic pane. It is actually a 4DGL code that allows the user to customise the behaviour of a Genie program. The Magic Object handles the WRITE_MAGIC_BYTES messages coming from the host. The user, for instance, can create a Magic Object that waits for a certain sequence of bytes coming from the host. The sequence can contain an instruction byte (e.g. a file size request) and a null-terminated 8.3 format filename (e.g. "datalog1.txt"). Upon receiving this the display module will get the size of the file and send it back to the host. The ViSi-Genie example project "**FileAccess.4DGenie**" is an implementation of the above application.

The application note **ViSi-Genie Magic How to Append to a File Arduino** presents and discusses a sketch for an Arduino program that sends a WRITE_MAGIC_BYTES message to and receives and handles REPORT_MAGIC_EVENT_BYTES messages coming from the display module. The WRITE_MAGIC_BYTES message contains the data to be appended and

the name of the file to which data will be appended and the REPORT_MAGIC_EVENT_BYTES message contains a success- or failure-indicator byte.

In this application note, a sketch for an Arduino program that performs a file size request operation, a file append operation, and then another file size request operation is discussed. This sequence was used to show that the file size request operation is indeed functioning. To understand the file append part of the operation, the reader is advised to refer to the application notes **ViSi-Genie Magic How to Append to a File** and **ViSi-Genie Magic How to Append to a File Arduino**. This application note will focus only on the file size request aspect of the operation.

Similar to the file append operation, the file size request operation is implemented using a WRITE_MAGIC_BYTES and a REPORT_MAGIC_EVENT_BYTES message. First the Arduino host sends a WRITE_MAGIC_BYTES message to the display module. The WRITE_MAGIC_BYTES message will contain the name of the file whose size is being requested for. The display module then performs the requested operation and sends a REPORT_MAGIC_EVENT_BYTES message back to the Arduino host. The REPORT_MAGIC_EVENT_BYTES message will contain 4 bytes which represent the 32-bit size of the file. The Arduino host then receives and interprets the REPORT_MAGIC_EVENT_BYTES message.

Setup Procedure

At this point, it is assumed that the reader has an understanding of the project described in the application note **ViSi-Genie Magic File Size Request.** The remaining task now is to write the sketch for the Arduino program and upload the program to the Arduino host.

Note 1: The attached sketch was tested on an Arduino Uno. A software serial port was used for communicating with the display. The hardware serial port, Serial0, was used for communicating with the Serial Monitor of the Arduino IDE.

Note 2: The ViSi-Genie project is the example project "**FileAccess.4DGenie**" in Workshop (go to File -> Samples -> ViSi Genie Magic(Picaso/Diablo)).

Note 3: The file **"data.log"** needs to be copied to the uSD card of the display module. This filename is hard coded into the attached sketch. Open the attached folder "uSD card file" and copy the content to the uSD card.

Program the Arduino Host

A thorough understanding of the application note **ViSi-Genie Connecting a 4D Display to an Arduino Host** is required before attempting to proceed further beyond this point. **ViSi-Genie Connecting a 4D Display to an Arduino Host** provides all the basic information that a user needs to be able to get started with ViSi-Genie and Arduino. The following is a list of the topics discussed in **ViSi-Genie Connecting a 4D Display to an Arduino Host**.

- How to download and install the ViSi-Genie-Arduino library
- How to open a serial port for communicating with the display and how to set the baud rate
- **The genieAttachEventHandler() function – for Genie Magic, the equivalent of this is AttachMagicByteReader( )**
- How to reset the host and the display
- How to set the screen contrast
- How to send a text string
- The main loop
- Receiving data from the display
- The use of a non-blocking delay in the main loop
- How to change the status of an object
- How to know the status of an object
- **The user's event handler – for Genie Magic, the equivalent of this is myGenieMagicHandler()**

Discussion of any of these topics is avoided in other ViSi-Genie-Arduino application notes unless necessary. Users are encouraged to read **ViSi-Genie Connecting a 4D Display to an Arduino Host** first.

Understanding the Demo Sketch

Open the Arduino sketch "**FileSizeSS.ino**" and the PDF file "**programFlow.pdf**" attached to this document. The PDF file shows the flow of the Arduino sketch/program.

Attach the Magic Event Handler

```
genie.AttachMagicByteReader(myGenieMagicHandler);
```

This simply tells the ViSi-Genie Arduino library which function to call to process the magic events received from the display. The function to be called is a user-defined function.

Set Up the Project

Refer to the section "**Connect the Display Module to the Arduino Host**" of the application note "**ViSi-Genie Connecting a 4D Display to an Arduino Host**" for the following topics:

- Using the New 4D Arduino Adaptor Shield (Rev 2.00)
    - Definition of Jumpers and Headers
    - Default Jumper Settings
    - Change the Arduino Host Serial Port
    - Power the Arduino Host and the Display Separately
- Using the Old 4D Arduino Adaptor Shield (Rev 1)
- Connection Using Jumper Wires
- Changing the Serial port of the Genie Program
- Changing the Maximum String Length

Note 1: The attached Arduino sketch uses a software serial port to communicate with the display module and a hardware serial port to communicate with Serial Monitor running on the PC.

Expected Serial Terminal Output

Request for the Size of the File "DATA.LOG" and Print it

```
Sending open and read request for the file:
DATA.LOG
Received 5 bytes from magic object 0
message:0 0 1 54
File size: 340 byte/s


First file size inquiry successful.
```

Append Data to "DATA.LOG"

```
Sending the file append command.
17 bytes appended to the file.


File append operation successful!
```

Request Again for the Size of the File "DATA.LOG" and Print it

```
Received 5 bytes from magic object 0
message:0 0 1 65
File size:  357 byte/s        17 bytes added


Second file size inquiry successful.
Process ended.
```

Proprietary Information

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

4D Systems endeavours to ensure that the information in this document is correct and fairly stated but does not accept liability for any error or omission. The development of 4D Systems products and services is continuous and published information may not be up to date. It is important to check the current position with 4D Systems.

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

Disclaimer of Warranties & Limitation of Liability

4D Systems makes no warranty, either expresses or implied with respect to any product, and specifically disclaims all other warranties, including, without limitation, warranties for merchantability, non-infringement and fitness for any particular purpose.

Information contained in this publication regarding device applications and the like is provided only for your convenience and may be superseded by updates. It is your responsibility to ensure that your application meets with your specifications.

In no event shall 4D Systems be liable to the buyer or to any third party for any indirect, incidental, special, consequential, punitive or exemplary damages (including without limitation lost profits, lost savings, or loss of business opportunity) arising out of or relating to any product or service provided or to be provided by 4D Systems, or the use or inability to use the same, even if 4D Systems has been advised of the possibility of such damages.

4D Systems products are not fault tolerant nor designed, manufactured or intended for use or resale as on line control equipment in hazardous environments requiring fail – safe performance, such as in the operation of nuclear facilities, aircraft navigation or communication systems, air traffic control, direct life support machines or weapons systems in which the failure of the product could lead directly to death, personal injury or severe physical or environmental damage ('High Risk Activities'). 4D Systems and its suppliers specifically disclaim any expressed or implied warranty of fitness for High Risk Activities.

Use of 4D Systems' products and devices in 'High Risk Activities' and in any other application is entirely at the buyer's risk, and the buyer agrees to defend, indemnify and hold harmless 4D Systems from any and all damages, claims, suits, or expenses resulting from such use. No licenses are conveyed, implicitly or otherwise, under any 4D Systems intellectual property rights.