APPLICATION NOTES

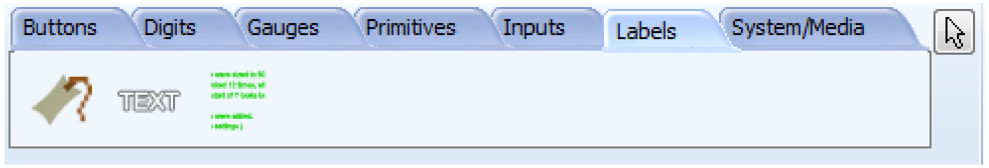
4D SYSTEMS
TURNING TECHNOLOGY INTO ART

ViSi-Genie Labels, Texts, and Strings

DOCUMENT DATE:	**29th April 2019**
DOCUMENT REVISION:	**1.1**

Description

This Application Note explores the possibilities provided by ViSi-Genie for the **Labels** objects:



This application note requires:

- Workshop 4 has been installed according to the document Workshop 4 Installation;
- The user is familiar with the Workshop 4 environment and with the fundamentals of ViSi-Genie, as described in Workshop 4 User Guide and ViSi-Genie User Guide;
- 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 discussed in these recommended application notes.

Three ViSi-Genie projects are provided as examples to help you along this application note.

Content

Application Overview

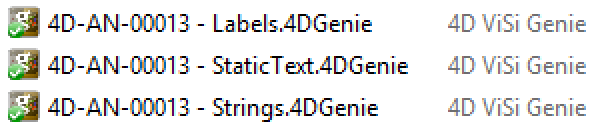
It is often difficult to design a graphical display without being able to see the immediate results of the application code. ViSi-Genie is the perfect software tool that allows the user to see the instant results of their desired graphical layout with this large selection of gauges and meters that can simply be dragged and dropped onto the simulated module display.

- Label
- Static text
- Strings

Each object can have properties edited and at the click of a button, all relevant code is produced in the user program. Each feature of ViSi-Genie will be outlined with examples below.

Setup Procedure

This application note comes with a zip file which contains three ViSi-Genie projects.


4D-AN-00013 - Labels.4DGenie 4D ViSi Genie
4D-AN-00013 - StaticText.4DGenie 4D ViSi Genie
4D-AN-00013 - Strings.4DGenie 4D ViSi Genie

For instructions on how to launch Workshop 4, how to open a ViSi-Genie project, and how to change the target display, kindly refer to the section "**Setup Procedure**" of the application note:

ViSi Genie Getting Started – First Project for Picaso Displays (for Picaso)
or
ViSi Genie Getting Started – First Project for Diablo16 Displays (for Diablo16).

Create a New Project

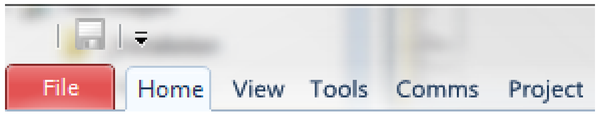
Create a New Project

For instructions on how to create a new ViSi-Genie project, please refer to the section "**Create a New Project**" of the application note

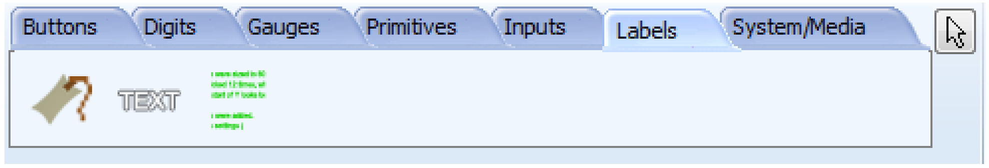
ViSi Genie Getting Started – First Project for Picaso Displays (for Picaso)
or
ViSi Genie Getting Started – First Project for Diablo16 Displays (for Diablo16).

Simulation Procedure

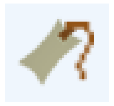
Select the **Home** menu to display the objects:



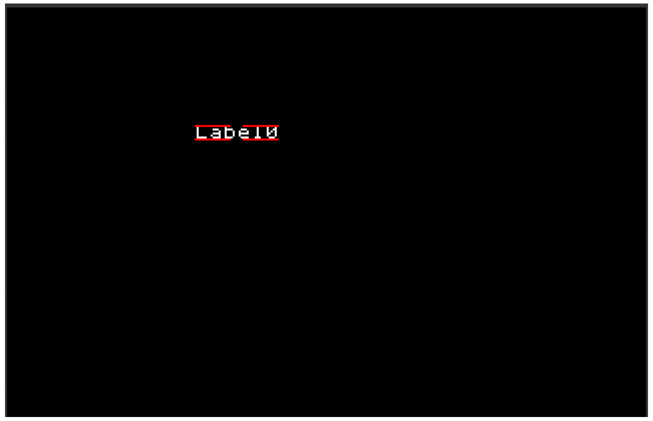
The **Labels** objects are located on the Labels pane:



To add an object, first click on the desired icon, here start with the first one, the **Label**…



…and then click on the WYSIWYG screen to place it.



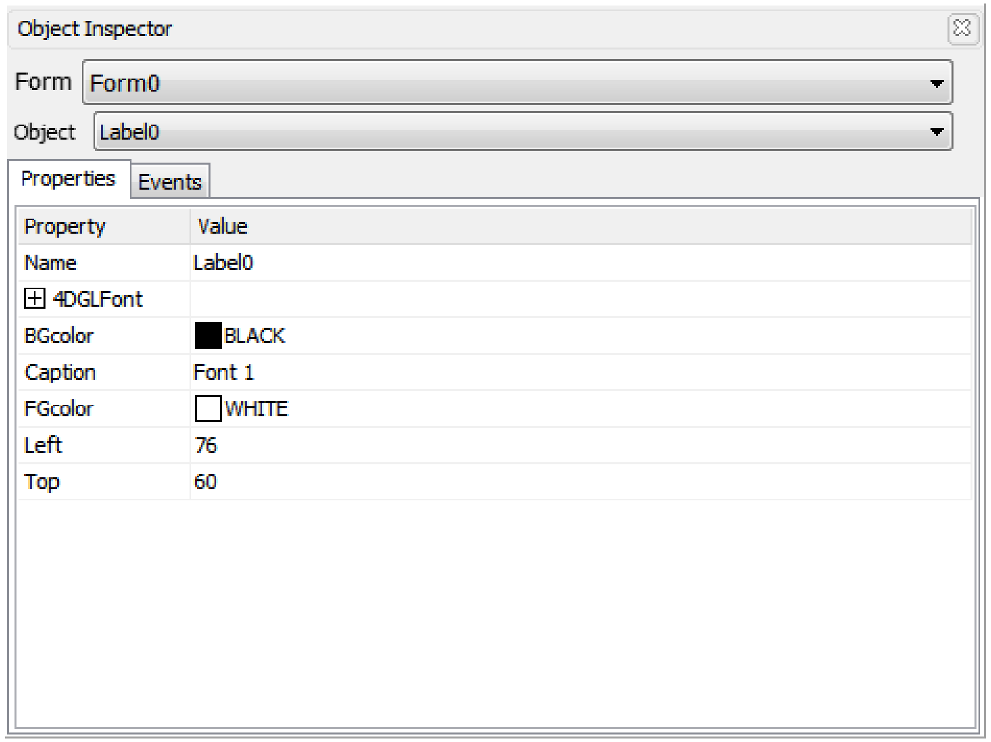
Label Options

You can load the example…

Example: 4D-AN-00013 – Labels

…or follow the procedures described hereafter.

The options are listed on the Object Inspector:



Object Inspector

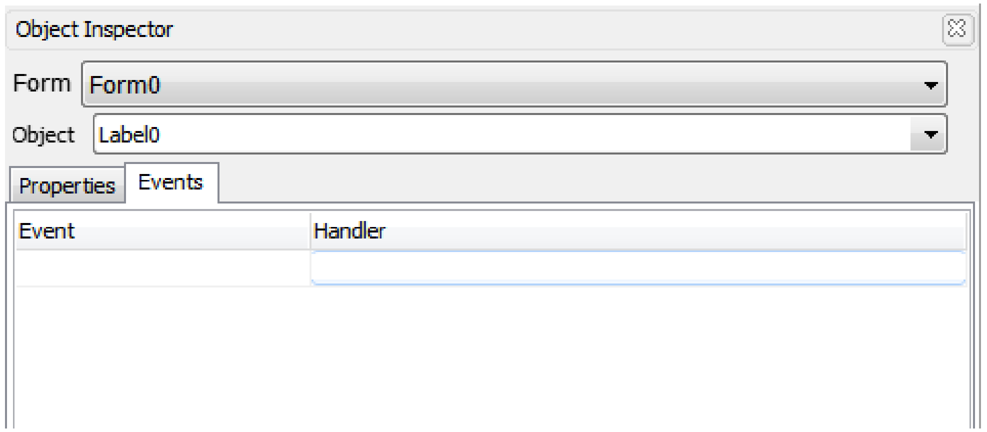
Form: Form0

Object: Label0

Properties | Events

Property	Value
Name	Label0
⊞ 4DGLFont	
BGcolor	■ BLACK
Caption	Font 1
FGcolor	□ WHITE
Left	76
Top	60

The **Label** object has no event.



Object Inspector

Form: Form0

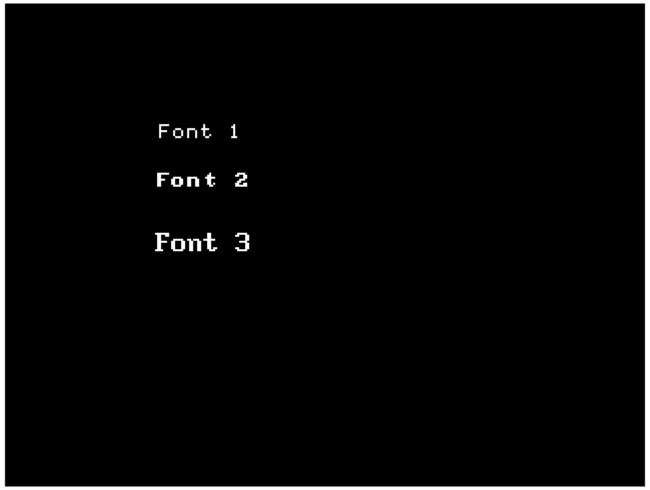
Object: Label0

Properties | Events

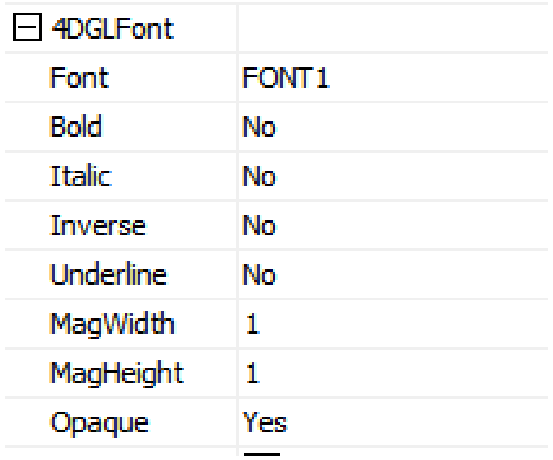
Event	Handler

Font Options

Three fonts are available: small FONT1, medium FONT2 and large FONT3. Those fonts are provided by the screen module.



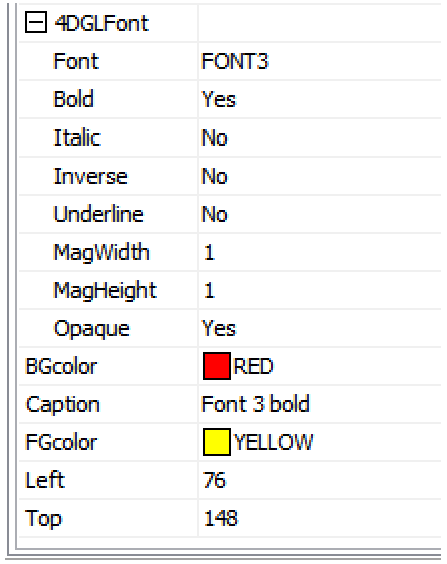
The font can be set to bold, italic, underline, inverse, opaque.



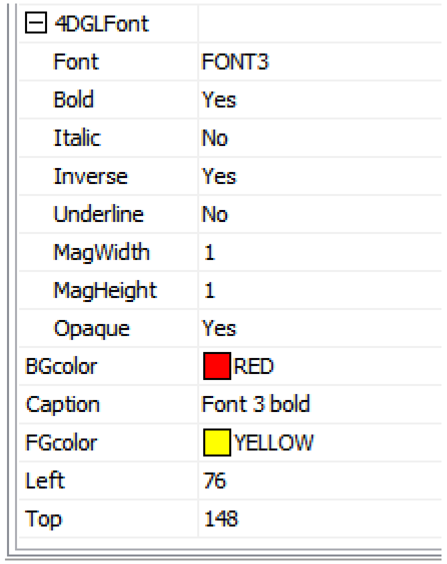
4DGLFont	
Font	FONT1
Bold	No
Italic	No
Inverse	No
Underline	No
MagWidth	1
MagHeight	1
Opaque	Yes

Colour Options

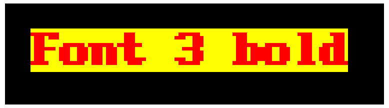
Front colour and background colour can be selected. Set **Opaque** to yes to display the background colour. Set **Inverse** to yes to switch the front and background colours.

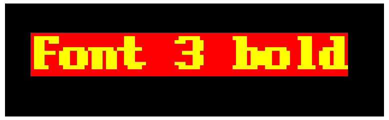


⊟ 4DGLFont	
Font	FONT3
Bold	Yes
Italic	No
Inverse	No
Underline	No
MagWidth	1
MagHeight	1
Opaque	Yes
BGcolor	🟥 RED
Caption	Font 3 bold
FGcolor	🟨 YELLOW
Left	76
Top	148



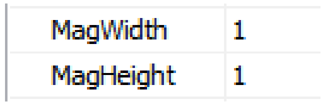
⊟ 4DGLFont	
Font	FONT3
Bold	Yes
Italic	No
Inverse	Yes
Underline	No
MagWidth	1
MagHeight	1
Opaque	Yes
BGcolor	🟥 RED
Caption	Font 3 bold
FGcolor	🟨 YELLOW
Left	76
Top	148



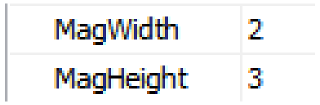


Magnification Options

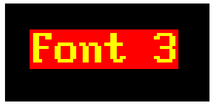
The font can be adjusted both vertically and horizontally. By default, magnification is 1 on both axes.

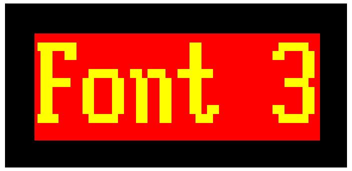


MagWidth	1
MagHeight	1



MagWidth	2
MagHeight	3





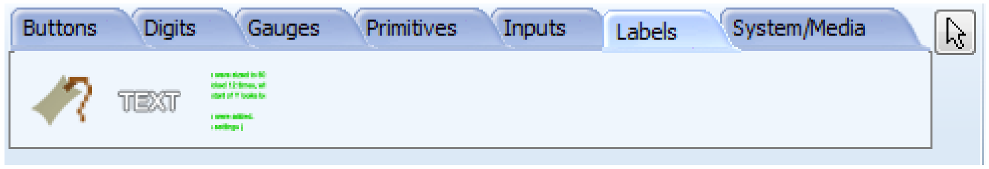
StaticText Options

You can load the example…

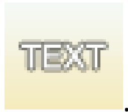
Example: 4D-AN-00013 – StaticText

…or follow the procedures described hereafter.

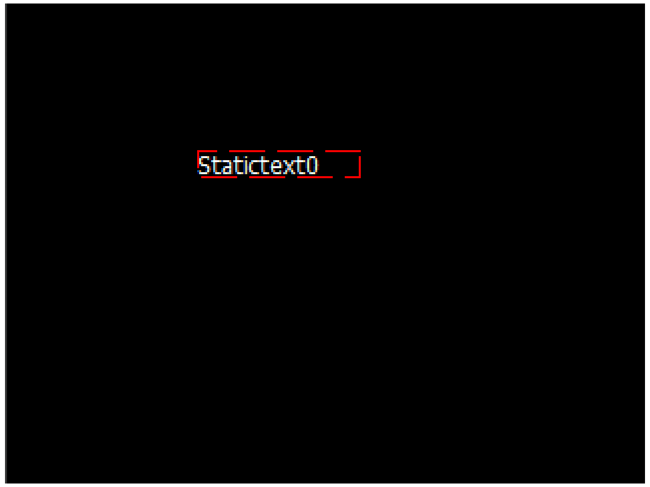
To add a StaticText object, select the **Labels** pane…



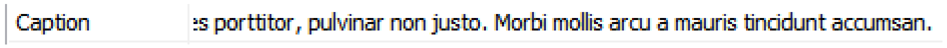
…click on the **StaticText** icon…



…and then click on the WYSIWYG screen to place it:



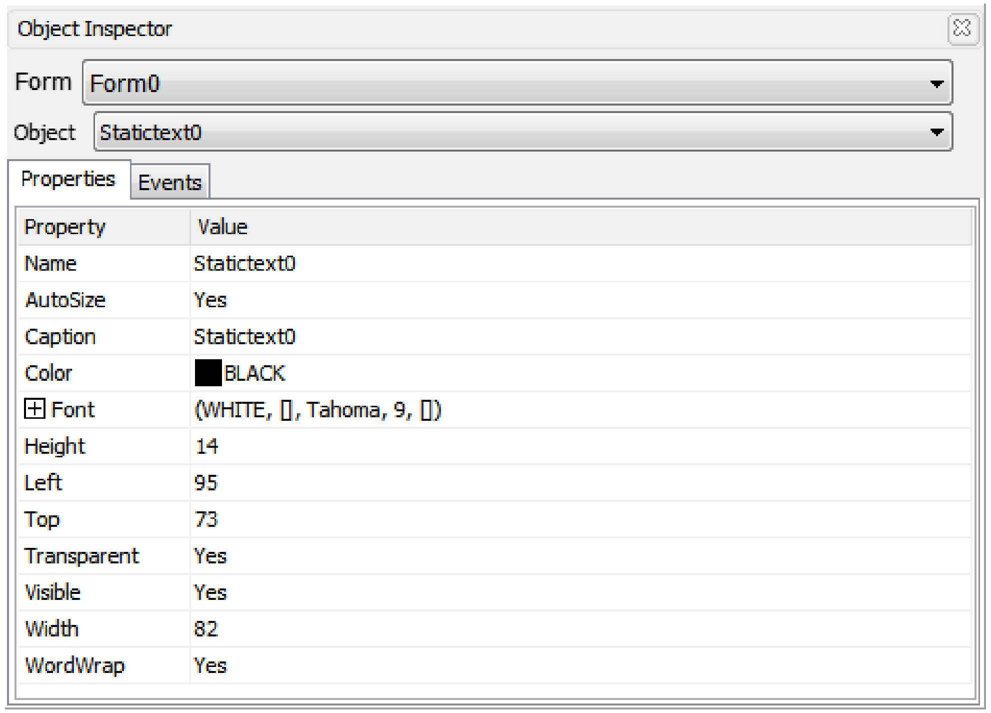
Enter the text on the **Caption** field:



Caption	:s porttitor, pulvinar non justo. Morbi mollis arcu a mauris tincidunt accumsan.

Use **\n** for line-feed.

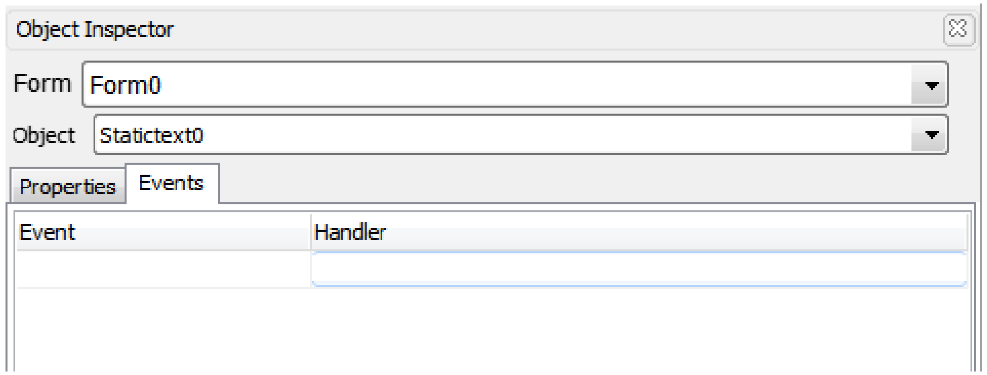
The options are listed on the Object Inspector:



Form: Form0
Object: Statictext0

Property	Value
Name	Statictext0
AutoSize	Yes
Caption	Statictext0
Color	■ BLACK
⊞ Font	(WHITE, [], Tahoma, 9, [])
Height	14
Left	95
Top	73
Transparent	Yes
Visible	Yes
Width	82
WordWrap	Yes

The **StaticText** object has no event.

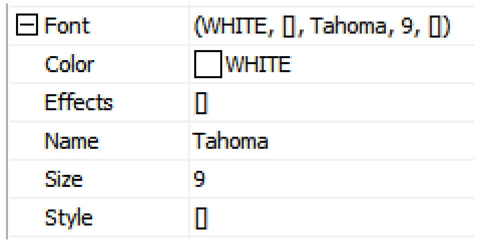


Form: Form0
Object: Statictext0

Event	Handler

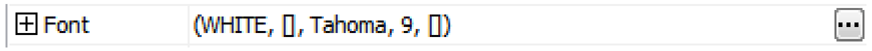
Font Options

The fonts used by the **StaticText** object include all the fonts from Windows.



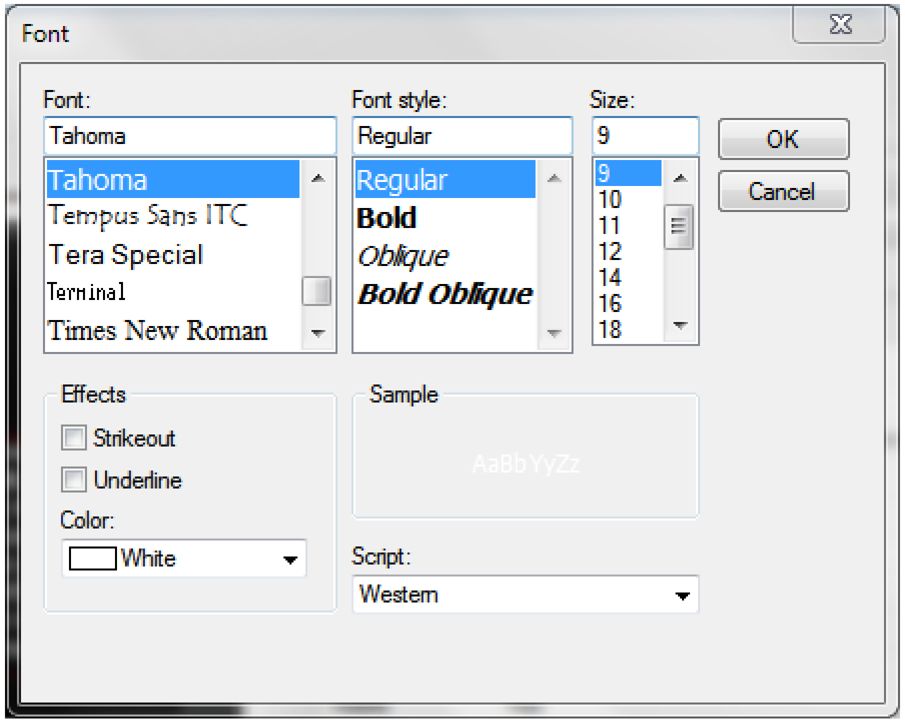
⊟ Font	(WHITE, [], Tahoma, 9, [])
Color	☐ WHITE
Effects	[]
Name	Tahoma
Size	9
Style	[]

Click on the … symbol on the right…

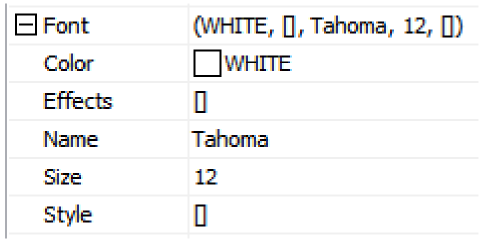


⊞ Font	(WHITE, [], Tahoma, 9, []) …

…to display the Font window:

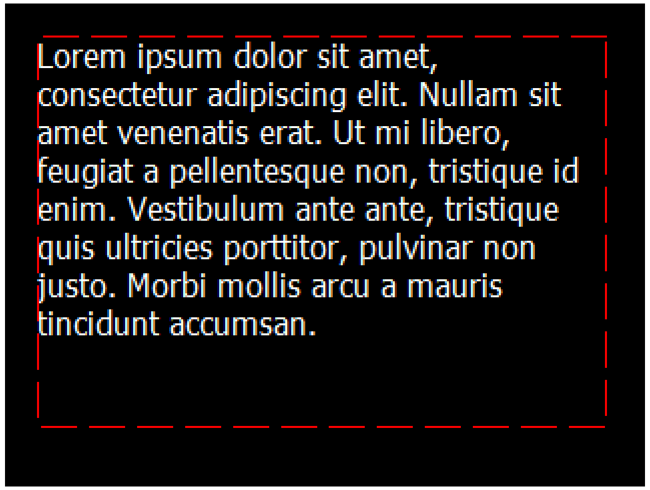


Select the font and change the size. Choosing Tahoma 12…



⊟ Font	(WHITE, [], Tahoma, 12, [])
Color	☐ WHITE
Effects	[]
Name	Tahoma
Size	12
Style	[]

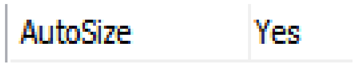
…to obtain the following result:



Lorem ipsum dolor sit amet, consectetur adipiscing elit. Nullam sit amet venenatis erat. Ut mi libero, feugiat a pellentesque non, tristique id enim. Vestibulum ante ante, tristique quis ultricies porttitor, pulvinar non justo. Morbi mollis arcu a mauris tincidunt accumsan.

Auto-Size Option

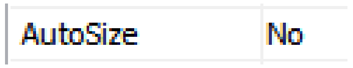
By default, the auto-size option is set to Yes.



AutoSize Yes

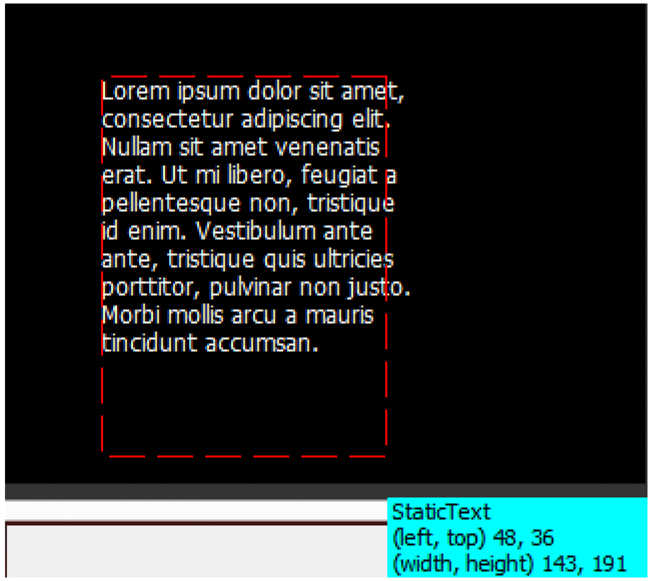
When auto-size is activated, the size of the static-text object is automatically adjusted for the text.

Setting **AutoSize** to No allows using the mouse to change the width and height of the static text object.



AutoSize No

Click on the red dotted line and resize the object. The text is then automatically reformatted.

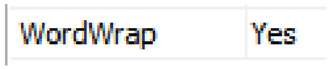


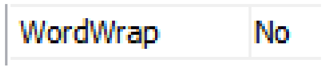
Lorem ipsum dolor sit amet, consectetur adipiscing elit. Nullam sit amet venenatis erat. Ut mi libero, feugiat a pellentesque non, tristique id enim. Vestibulum ante ante, tristique quis ultricies porttitor, pulvinar non justo. Morbi mollis arcu a mauris tincidunt accumsan.

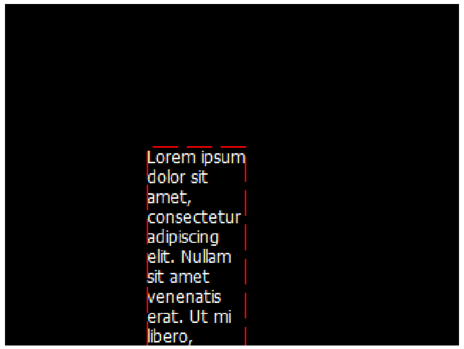
StaticText
(left, top) 48, 36
(width, height) 143, 191

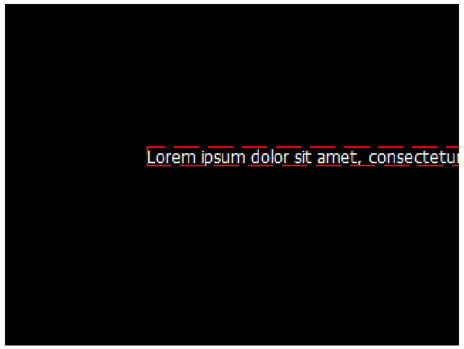
Word-Wrap Option

The word-wrap option automatically includes carriage returns.


WordWrap Yes


WordWrap No


Lorem ipsum
dolor sit
amet,
consectetur
adipiscing
elit. Nullam
sit amet
venenatis
erat. Ut mi
libero,

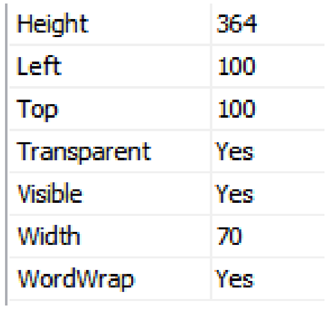

Lorem ipsum dolor sit amet, consectetu

To fit the text on the screen, either

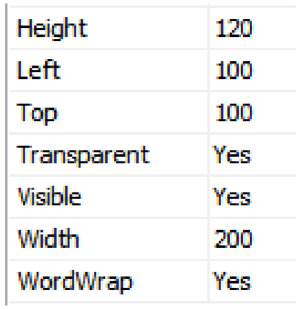
- Adjust the **Width** and the **Height** of the **StaticText** object manually if **AutoSize** is on,

or

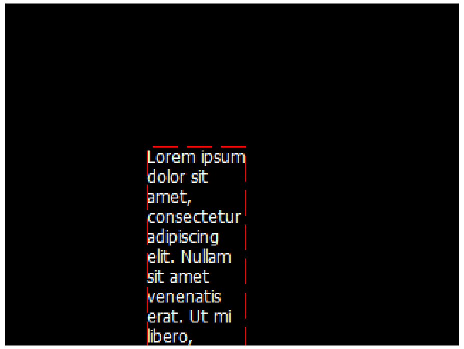
- Set **AutoSize** to no and use the mouse:



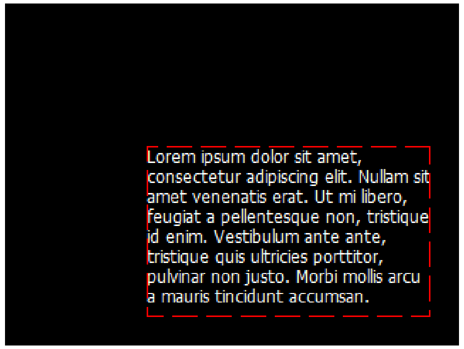
Height	364
Left	100
Top	100
Transparent	Yes
Visible	Yes
Width	70
WordWrap	Yes



Height	120
Left	100
Top	100
Transparent	Yes
Visible	Yes
Width	200
WordWrap	Yes



Lorem ipsum
dolor sit
amet,
consectetur
adipiscing
elit. Nullam
sit amet
venenatis
erat. Ut mi
libero,



Lorem ipsum dolor sit amet,
consectetur adipiscing elit. Nullam sit
amet venenatis erat. Ut mi libero,
feugiat a pellentesque non, tristique
id enim. Vestibulum ante ante,
tristique quis ultricies porttitor,
pulvinar non justo. Morbi mollis arcu
a mauris tincidunt accumsan.

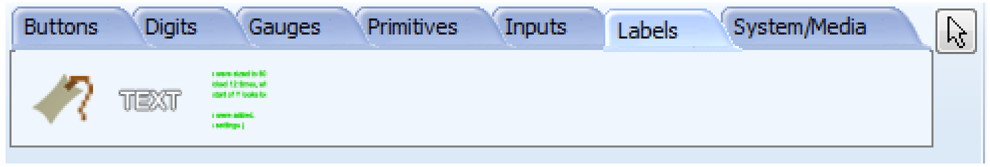
Strings Options

You can load the example…

Example: 4D-AN-00013 – String

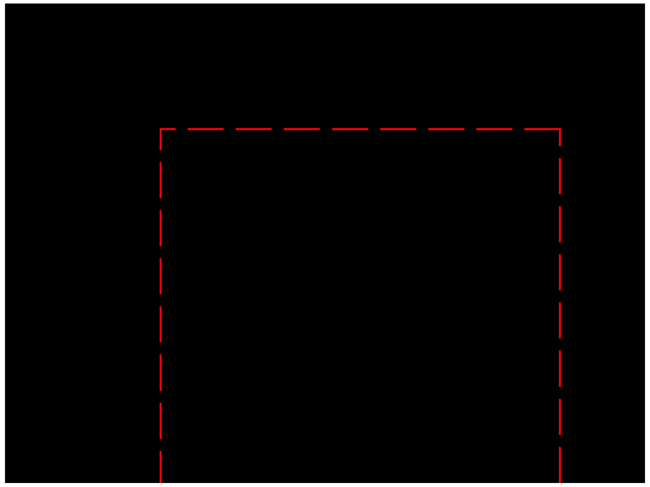
…or follow the procedures described hereafter.

To add a String object, select the **Labels** pane…



…click on the **Strings** icon…

…and then click on the WYSIWYG screen to place it:

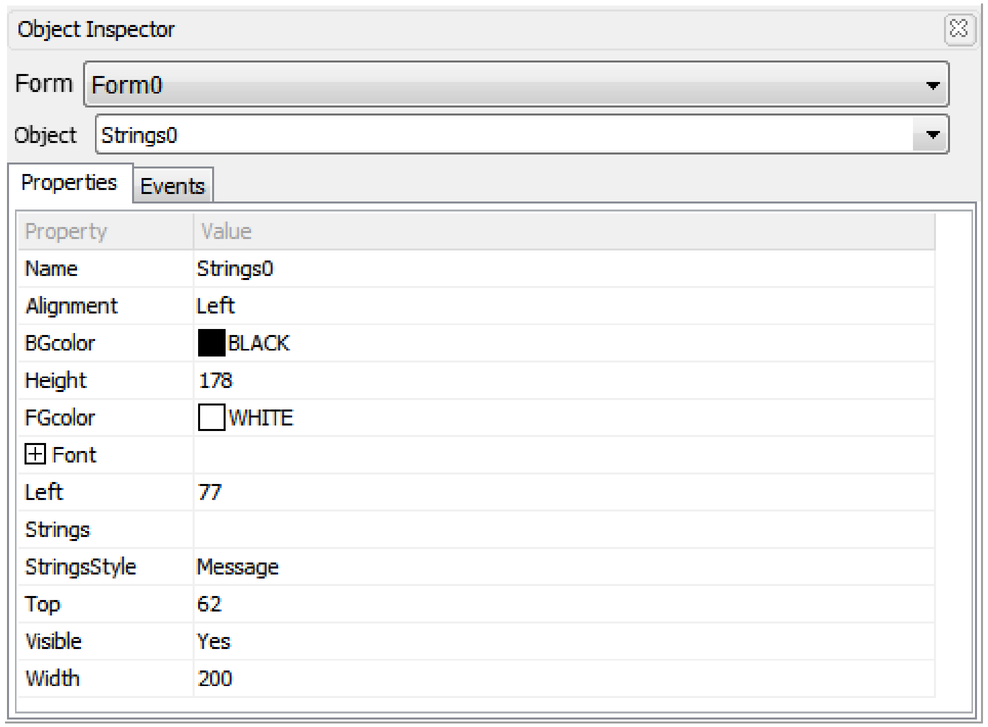


Predefined and Dynamic Strings

The area enclosed by the box can be thought of as the "container" or the "place holder" for that strings object. The text or string displayed inside this container can be classified into two – **predefined** strings and **dynamic** strings. Predefined strings are those that are created in Workshop during design time. These are actually saved into the uSD card and can later be retrieved during runtime. Dynamic strings, on the other hand, are those that are sent by a host to the display during runtime. A strings object (or container) can display a predefined string and a dynamic string, one at a time. There are separate commands for these.

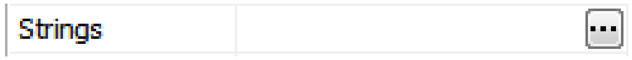
Predefined strings can be further classified by how they are displayed – message style and book style. For message style, the user creates a text composed of multiple parts (usually a part is composed of a line). During runtime, the strings object or container can be made to display the desired line. For book style, the user creates a text composed of multiple pages. During runtime, the strings container can be made to display the desired page. Displaying of message and book style strings will now be illustrated. Displaying of dynamic strings will be illustrated in the section "Debugger Output".

The options for a strings object are listed on the Object Inspector:

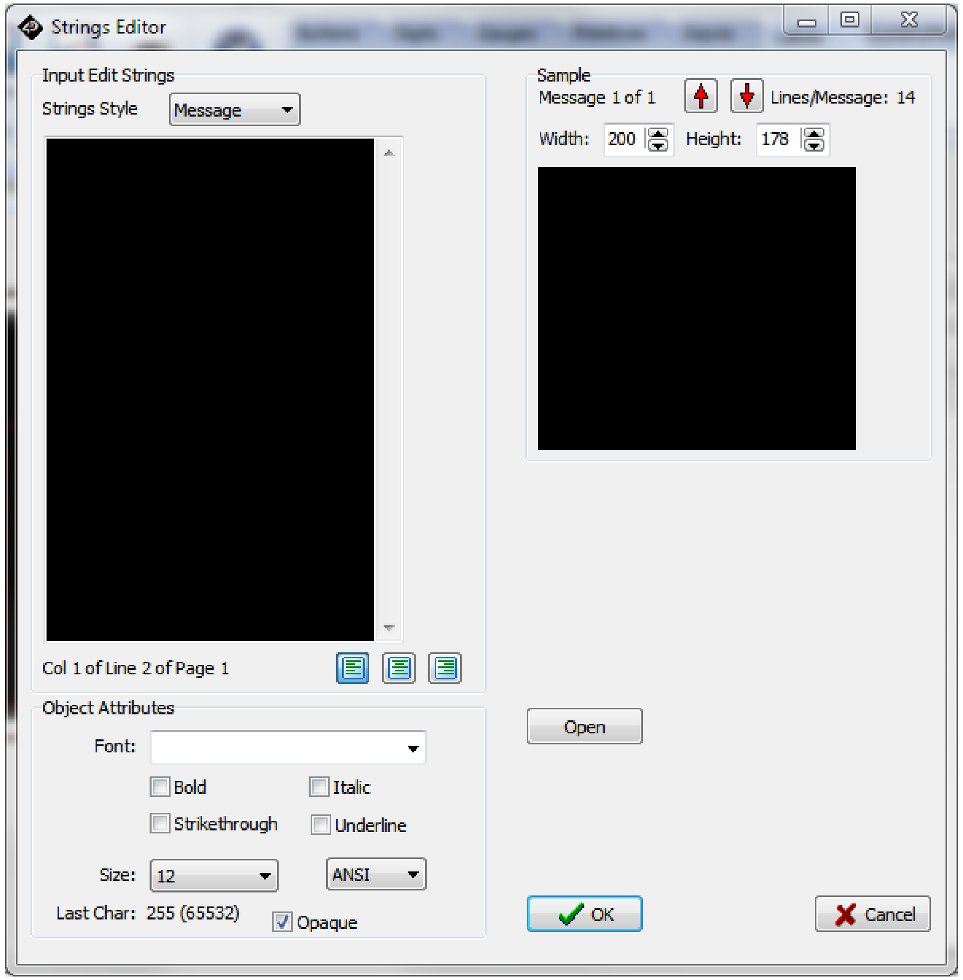


Property	Value
Name	Strings0
Alignment	Left
BGcolor	■ BLACK
Height	178
FGcolor	□ WHITE
⊞ Font	
Left	77
Strings	
StringsStyle	Message
Top	62
Visible	Yes
Width	200

The object is empty. To enter the text, select the line…

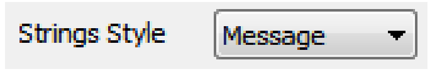


...and click on the ⋯ : a new window opens the **String Editor**:

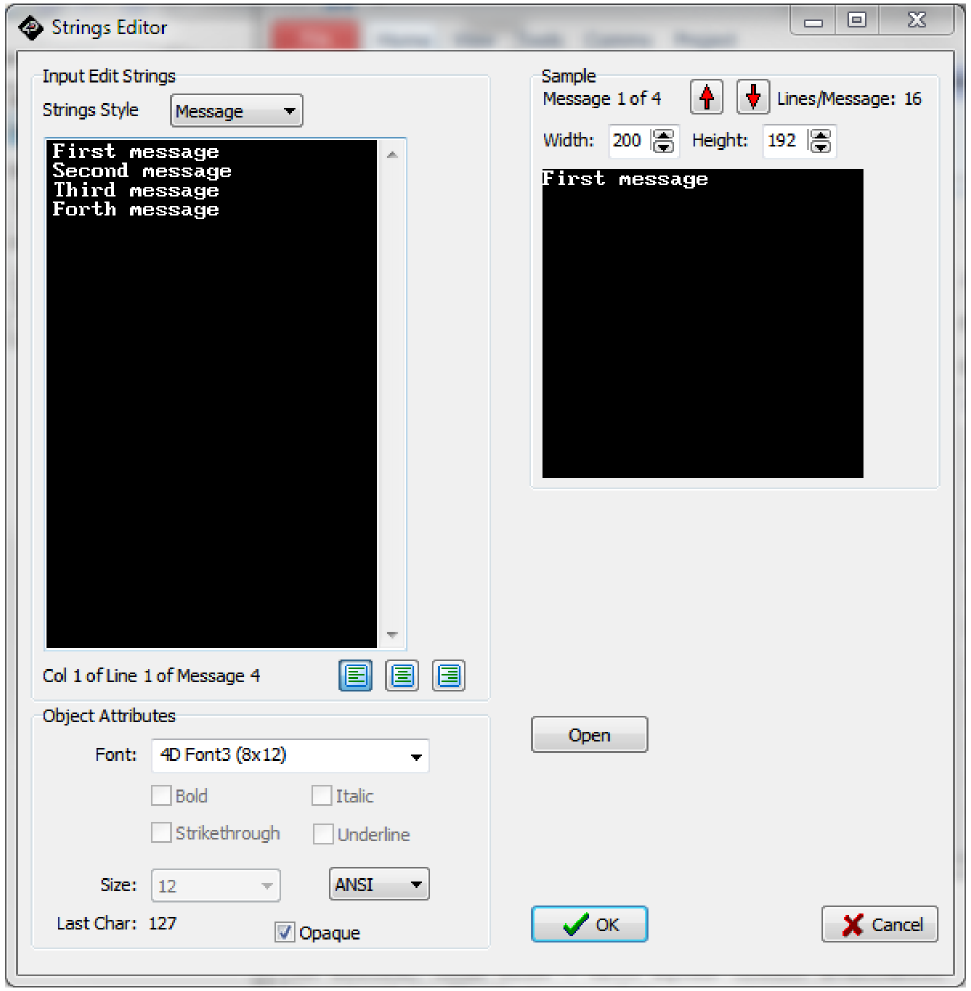


Message Style Option

On the Strings Editor, select the **Message** option:

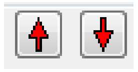


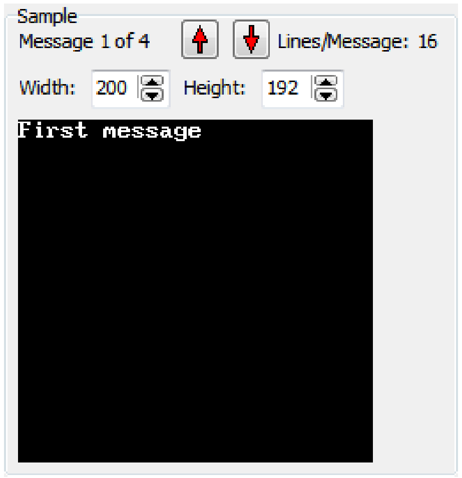
Enter each message, one message per line.

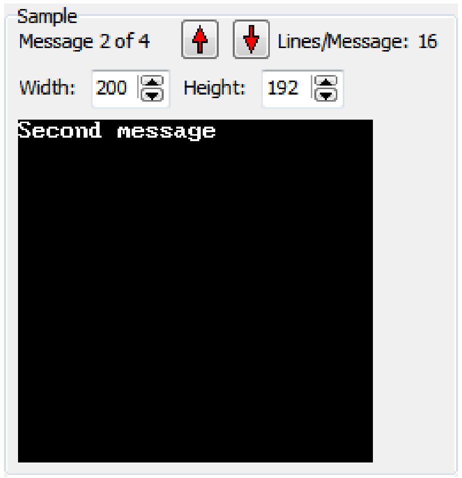


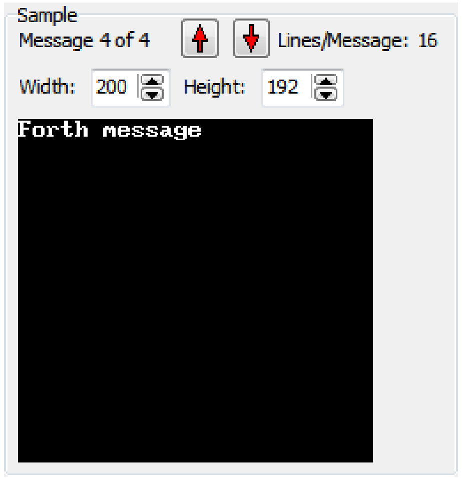
```
First message
Second message
Third message
Forth message
```

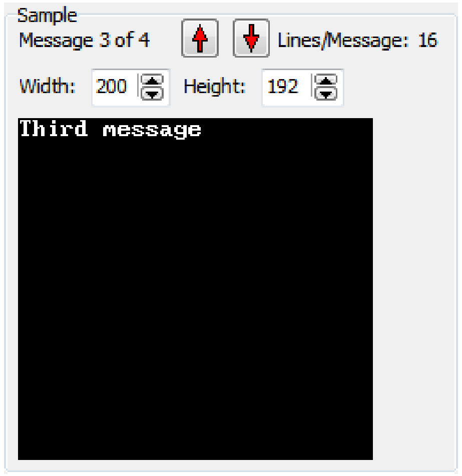
The messages can be monitored on the right part of the String Editor by using the arrows:



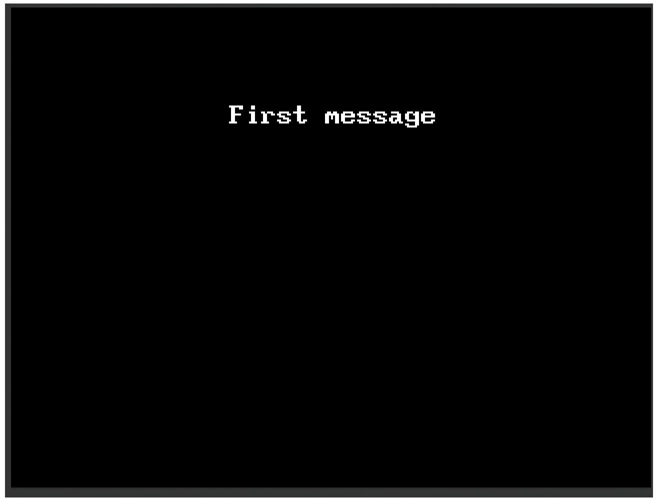




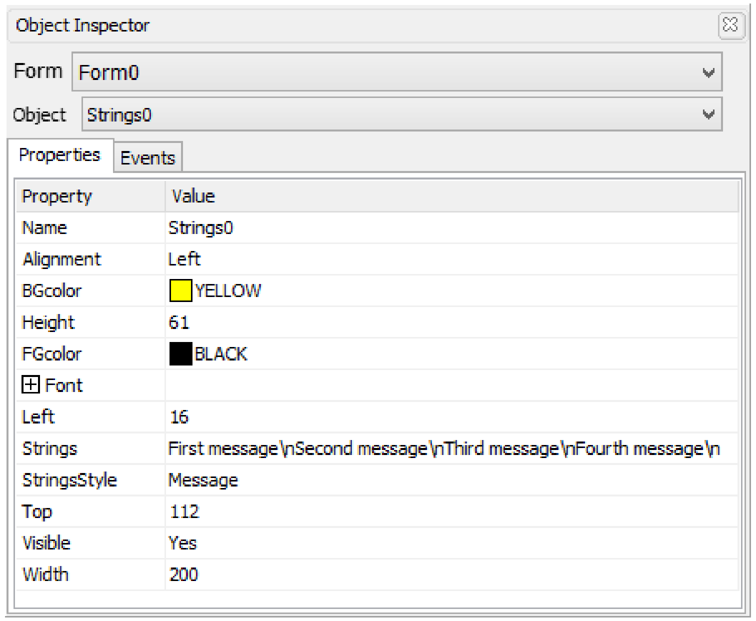




Each message is displayed separately:



The background and foreground colours can also be changed.



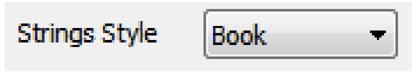
Form: Form0

Object: Strings0

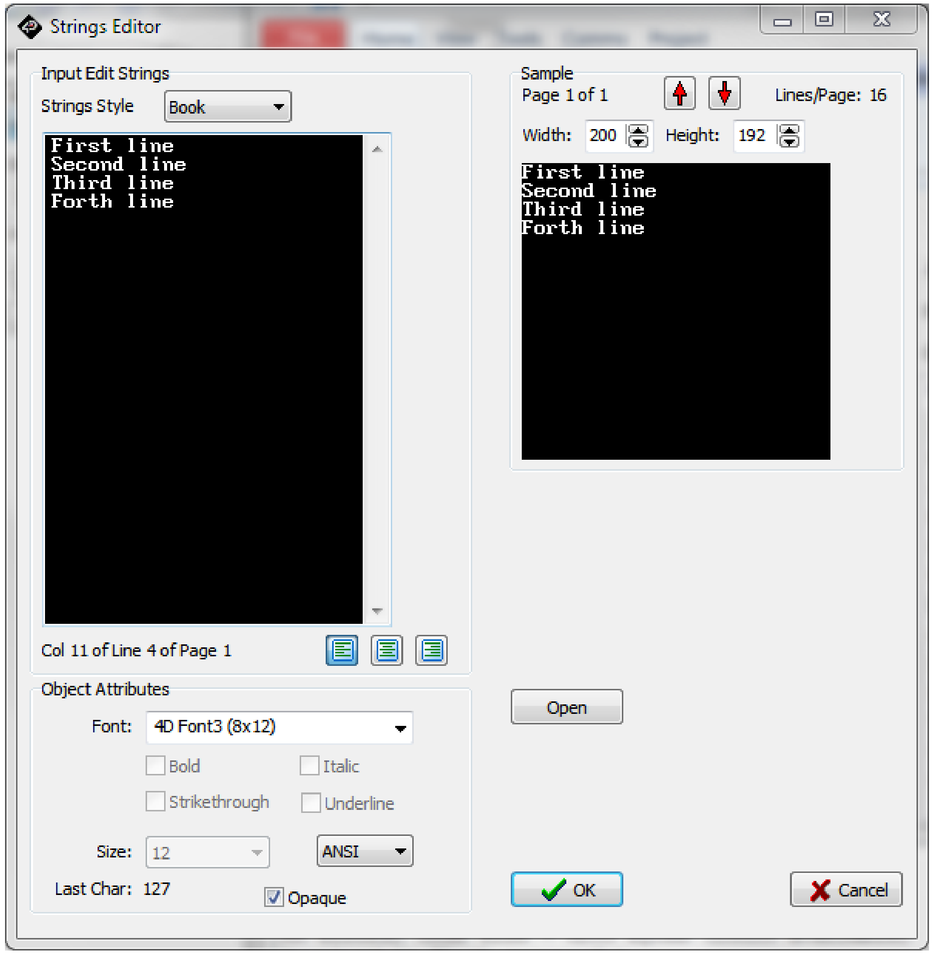
Property	Value
Name	Strings0
Alignment	Left
BGcolor	🟨 YELLOW
Height	61
FGcolor	⬛ BLACK
⊞ Font	
Left	16
Strings	First message\nSecond message\nThird message\nFourth message\n
StringsStyle	Message
Top	112
Visible	Yes
Width	200

Book Style Option

Create a strings object similar to Strings0 above. On the Strings Editor, select the **Book** option this time:

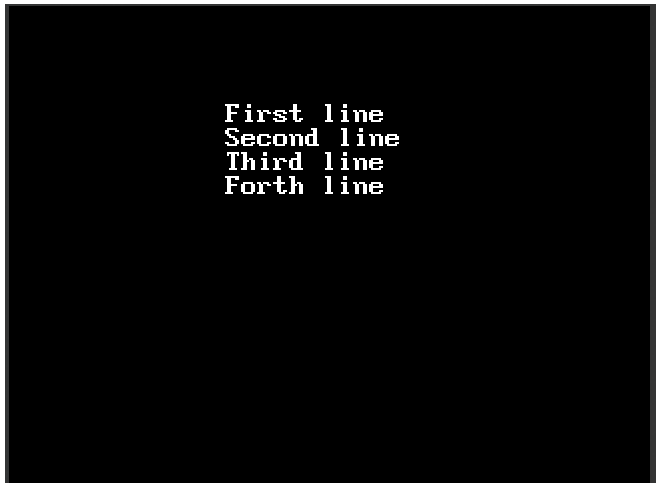


A book includes pages, each page with multiples lines:



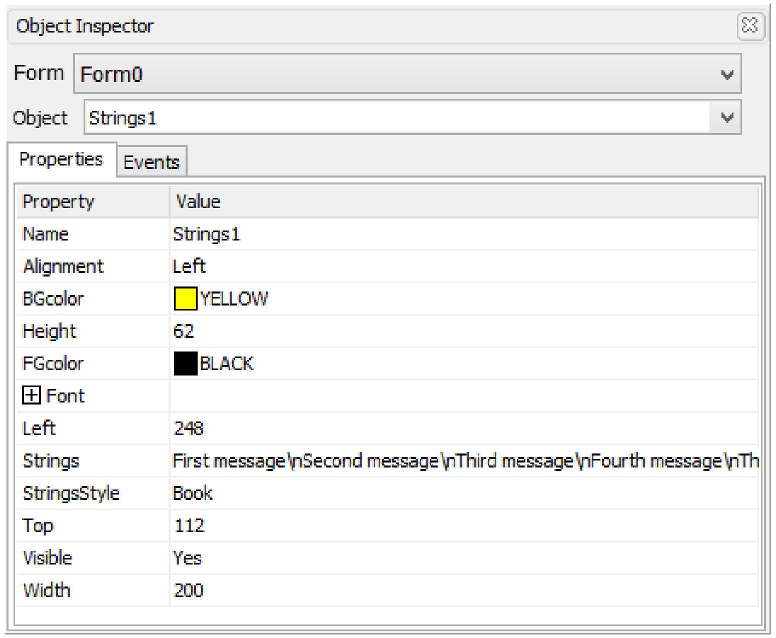
```
First line
Second line
Third line
Forth line
```

All the lines from the same page are displayed together:



```
First line
Second line
Third line
Forth line
```

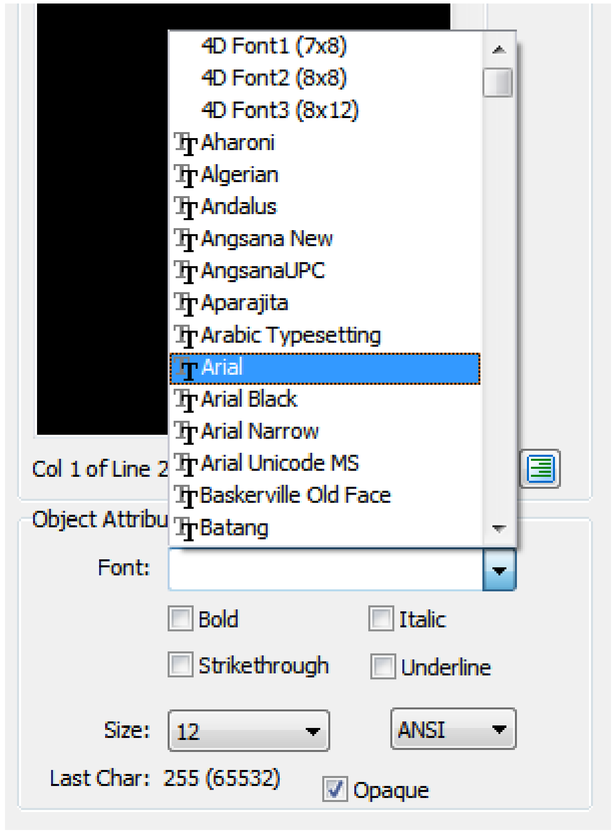
The background and foreground colours can also be changed.



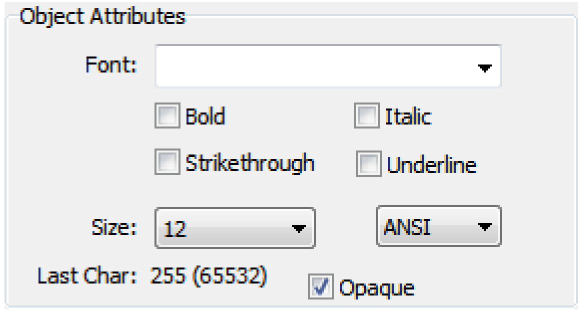
Property	Value
Name	Strings1
Alignment	Left
BGcolor	🟨YELLOW
Height	62
FGcolor	⬛BLACK
⊞ Font	
Left	248
Strings	First message\nSecond message\nThird message\nFourth message\nTh
StringsStyle	Book
Top	112
Visible	Yes
Width	200

Font Options

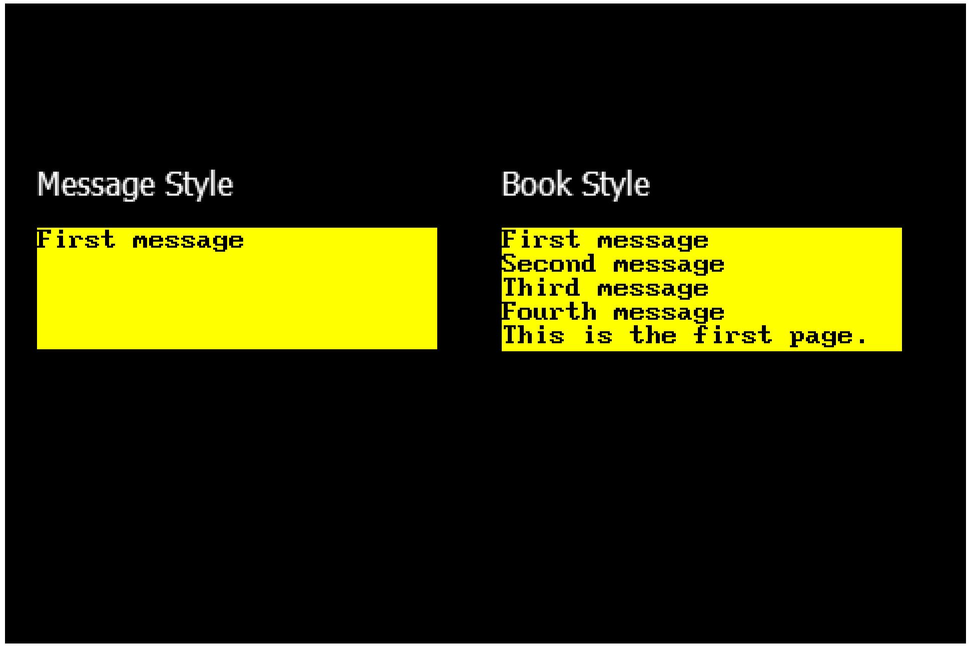
The font options offer a wide range of fonts, and include the fonts provided by the screen module and the fonts provided by Windows.



Attributes can be set, as the type ANSI or UNICODE:



With static text objects added as labels and the background and foreground colours changed, the final output is shown below.


Message Style

```
First message
```

Book Style

```
First message
Second message
Third message
Fourth message
This is the first page.
```

Build and Upload the Project

For instructions on how to build and upload a ViSi-Genie project to the target display, please refer to the section "**Build and Upload the Project**" of the application note

ViSi Genie Getting Started – First Project for Picaso Displays (for Picaso)
or
ViSi Genie Getting Started – First Project for Diablo16 Displays (for Diablo16).

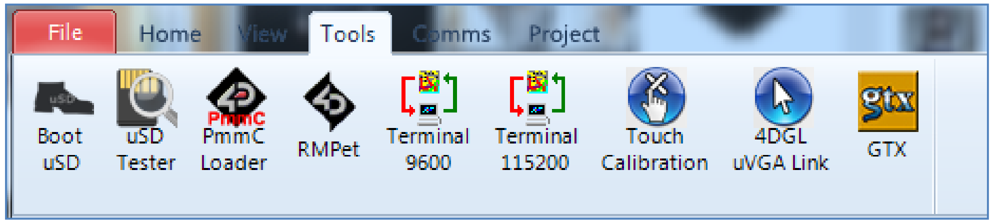
The uLCD-32PTU and/or the uLCD-35DT display modules are commonly used as examples, but the procedure is the same for other displays.

Debugger Output

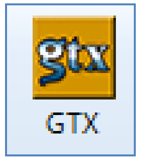
The GTX tool is used to send and receive messages to and from the display. It is an essential debugging tool.

Launch the Debugger

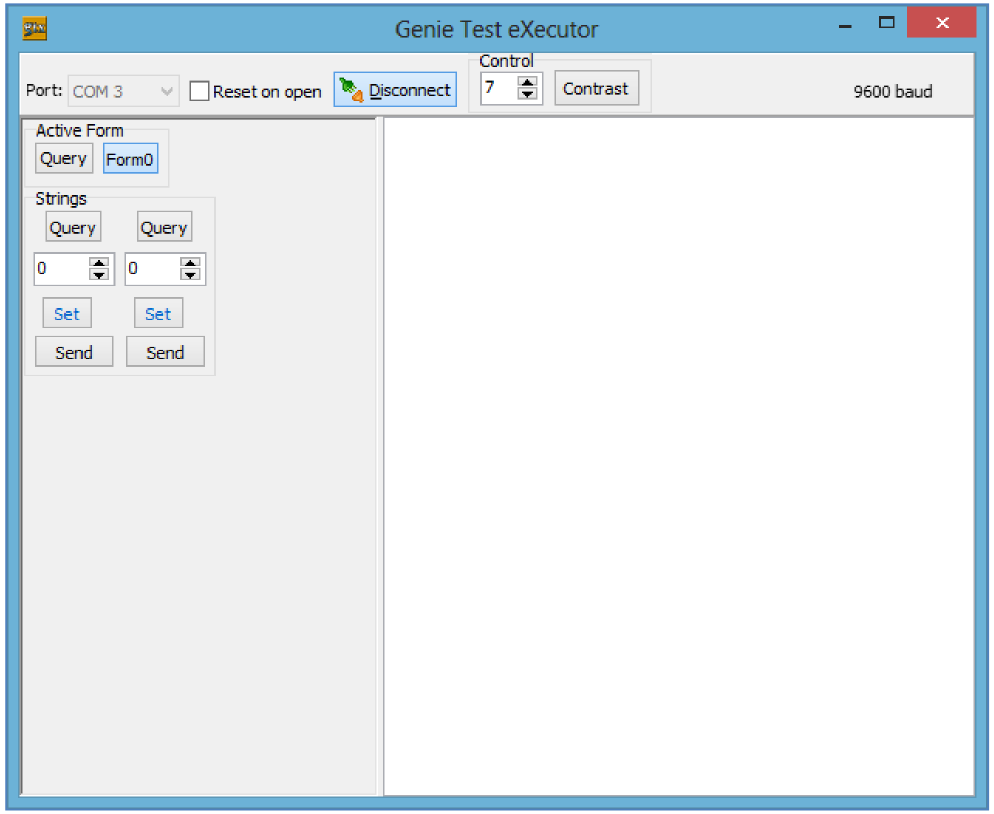
To launch the **Genie Test Executor** or GTX, select the **Tools** menu…



…and then click on the **GTX** button.

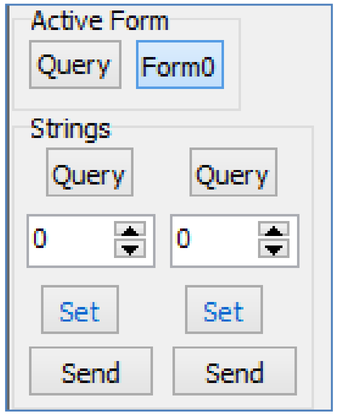


A new screen appears, with the form and objects we have defined previously, here from the Strings example:



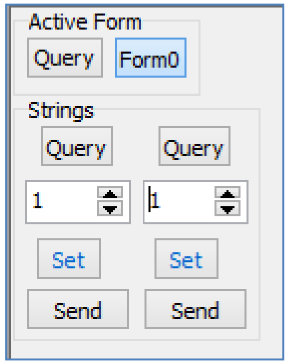
Use the Debugger

The debugger window provides all the controls for the strings object:

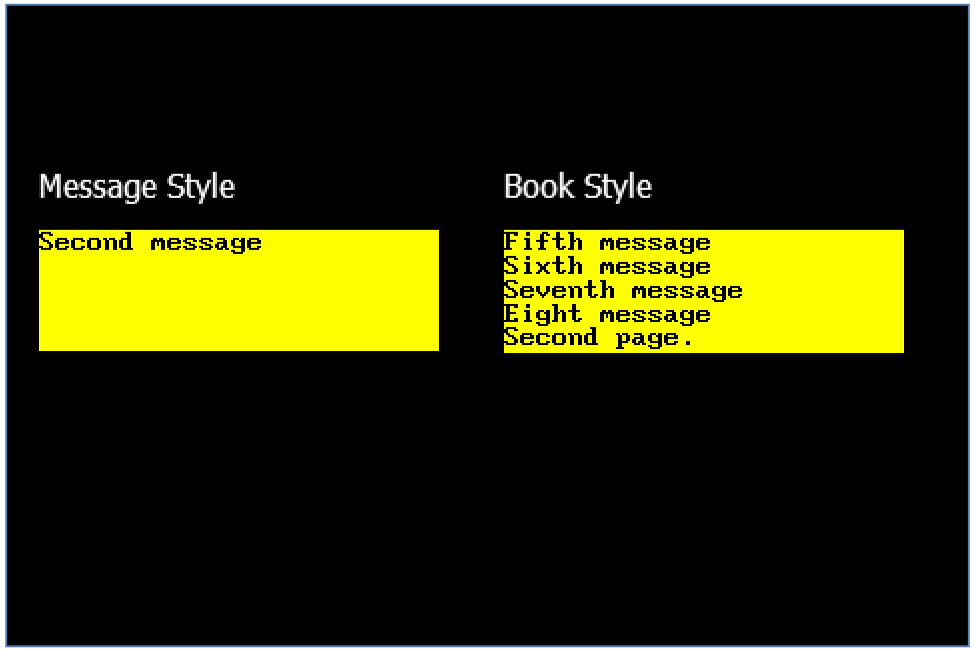


Display a Predefined String

To display the second message for Strings0 (message style) and to display the second page for Strings1, enter the value “1” into both fields as shown below and press on the “Set” buttons.



The appropriate line and page are now displayed.



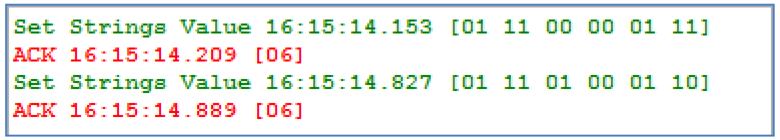
Message Style

Second message

Book Style

Fifth message
Sixth message
Seventh message
Eight message
Second page.

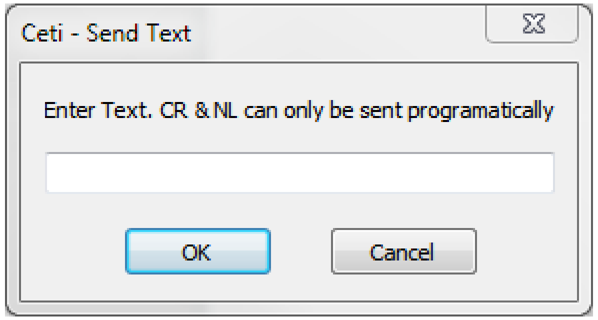
The right part of the GTX window displays the command sent and the acknowledgement **06**:



```
Set Strings Value 16:15:14.153 [01 11 00 00 01 11]
ACK 16:15:14.209 [06]
Set Strings Value 16:15:14.827 [01 11 01 00 01 10]
ACK 16:15:14.889 [06]
```

Send and Display a Dynamic String

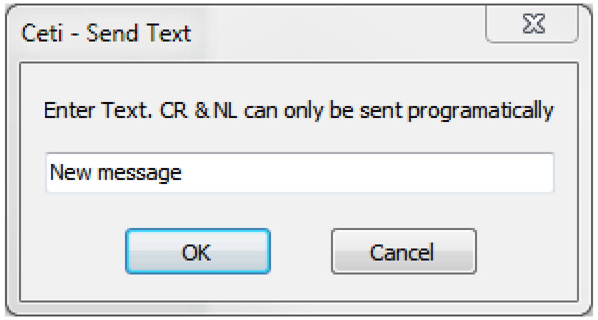
To send a dynamic string to the strings object String0, click on its "Send" button. A new **Send Text** window asks for the text to be sent…



Ceti - Send Text

Enter Text. CR & NL can only be sent programatically

Type the text in…

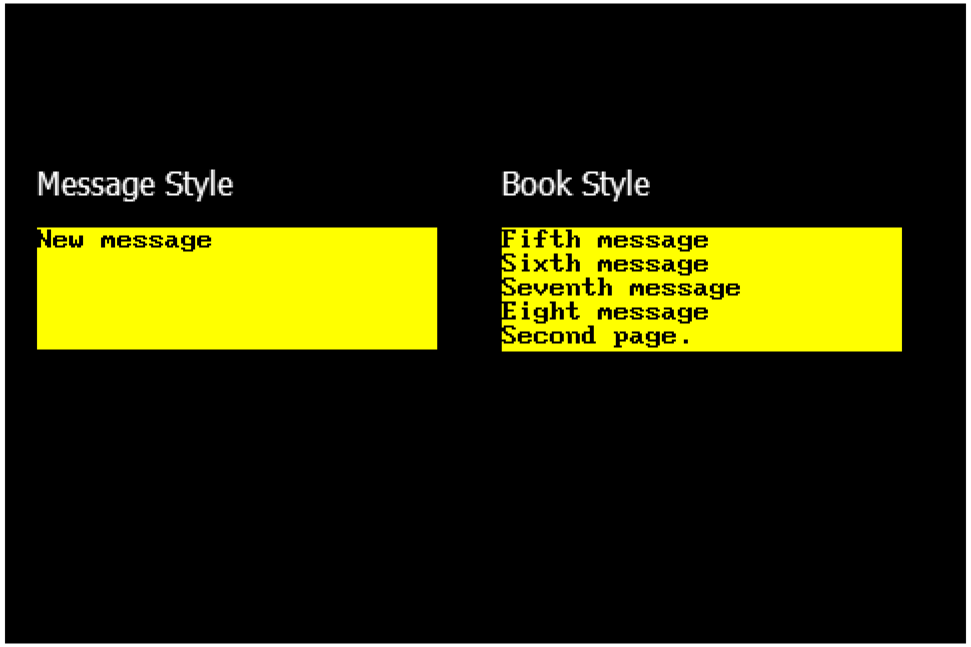


Ceti - Send Text

Enter Text. CR & NL can only be sent programatically

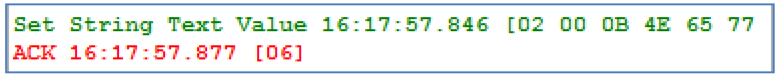
New message

…and press OK. The screen is updated accordingly:



This new message doesn't alter the recorded messages or the predefined strings.

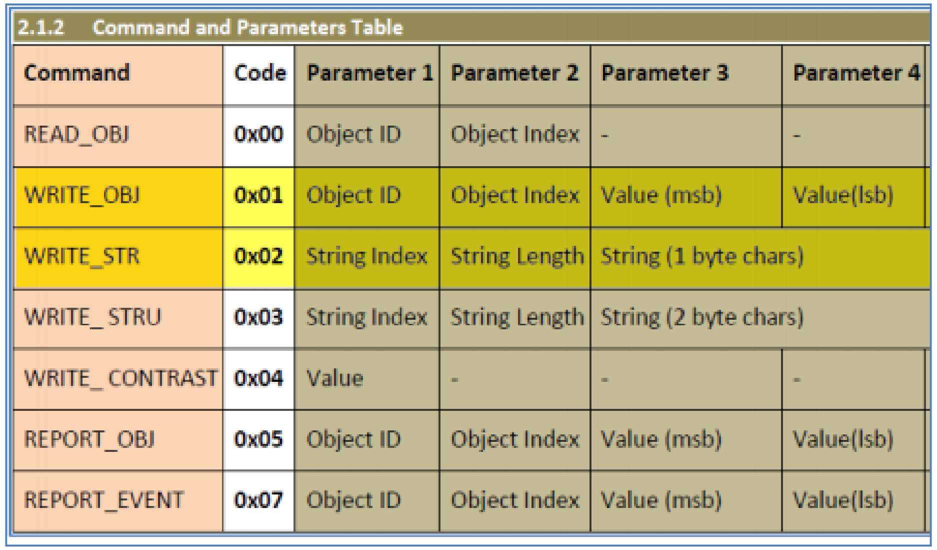
The right part of the GTX window displays the command sent and the acknowledgement **06**:



```
Set String Text Value 16:17:57.846 [02 00 0B 4E 65 77
ACK 16:17:57.877 [06]
```

Note the difference between the command for displaying a predefined string and that for displaying a dynamic string.

Referring to the ViSi-Genie Reference Manual, these two commands are highlighted below:



2.1.2 Command and Parameters Table

Command	Code	Parameter 1	Parameter 2	Parameter 3	Parameter 4
READ_OBJ	0x00	Object ID	Object Index	-	-
WRITE_OBJ	0x01	Object ID	Object Index	Value (msb)	Value(lsb)
WRITE_STR	0x02	String Index	String Length	String (1 byte chars)	
WRITE_ STRU	0x03	String Index	String Length	String (2 byte chars)	
WRITE_ CONTRAST	0x04	Value	-	-	-
REPORT_OBJ	0x05	Object ID	Object Index	Value (msb)	Value(lsb)
REPORT_EVENT	0x07	Object ID	Object Index	Value (msb)	Value(lsb)

Here is an additional information on strings object from the manual (section 3.2.5.2 Strings).

> *The first strings are displayed **initially**. Normally strings are set to predefined values, e.g. a value of 0 might display the string 'Hello There'. Using predefined values makes the most efficient use of the comms link and also minimizes the code required in your controller. In order to display a dynamically created string the user can send the Write String ASCII command message. The default maximum string length is 75, this can be changed in the Workshop options for Genie. For Unicode string objects Unicode strings can be sent, using the Write String Unicode command message. **CRs and LFs can be***

included and the user is responsible for the 'formatting' of the string, the formatting of strings in Workshop does not apply to dynamic strings.

Thus, the right -, centre -, or left - alignment of a predefined string when being formatted in the strings editor in Workshop does not apply to dynamic strings. Furthermore, a dynamic string does not automatically fit into the area defined by the strings container object. It will only be printed starting at the x and y coordinates of the strings container object. Formatting of predefined strings is set during design time, while formatting of dynamic strings is up to the host.

Another important point to consider is that when the program navigates between forms, a strings object on a certain form "remembers" the last predefined line or page of text it was made to display. However, if the string object is made to display a dynamic string and form navigation occurs, the string object will not remember anything. The host or program will have to make the strings object display another line or page of predefined text. The strings object will then remember this line or page, until the next dynamic string is received.