APPLICATION NOTES

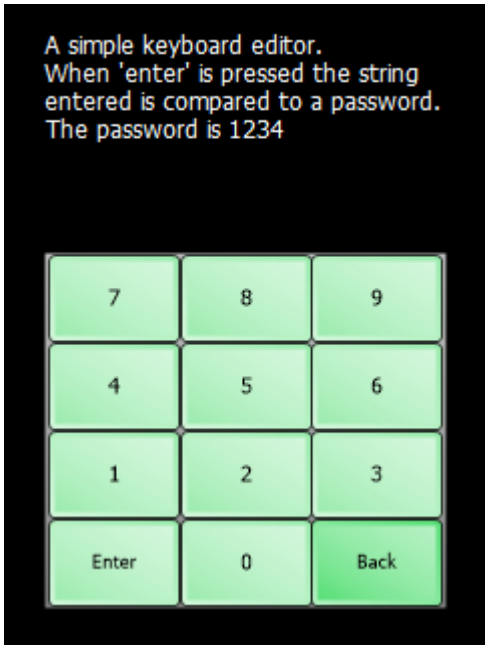
4D SYSTEMS
TURNING TECHNOLOGY INTO ART

ViSi–Genie Magic Keyboard Edit

DOCUMENT DATE: **21st May 2019**
DOCUMENT REVISION: **1.1**

Description

This application note primarily shows how to create a simple keyboard editor using magic Keyboard + ColorPicker Event Magic Object and other magic objects.



Note 1: The ViSi-Genie project for this application note is the demo "**KeyboardEdit**", which is found in Workshop. Go to the File menu -> Samples ->ViSi Genie Magic (Picaso/Diablo16) ->**KeyboardEdit.4DGenie**.

Note 2: Workshop Pro is needed for this application.

Before getting started, the following are required:

- Any of the following Picaso display modules:

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

and other superseded display modules which support the ViSi environment

- The target module can also be a Diablo16 display

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

In the past it was not possible to create an application in ViSi-Genie that could print(in a string object) the characters pressed on a keyboard object. This was possible only if a host like an Arduino board receives the keyboard's value and then prints it to a string object. The Keyboard + ColorPicker Event Magic Object was used to make this possible. The application shown in this document will prompt the user to input a password. If the password is correct it will proceed to a different form else if the input password is incorrect then it will print 'Fail' on the display.

Setup Procedure

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

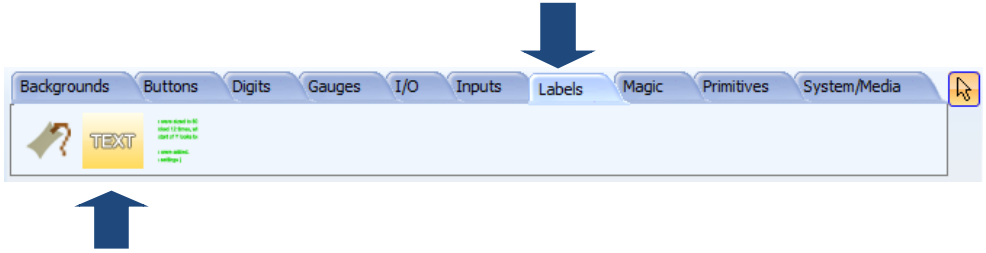
ViSi Genie Getting Started – First Project for Picaso Displays (for Picaso)
or
ViSi Genie Getting Started – First Project for Diablo16 Displays (for Diablo16).

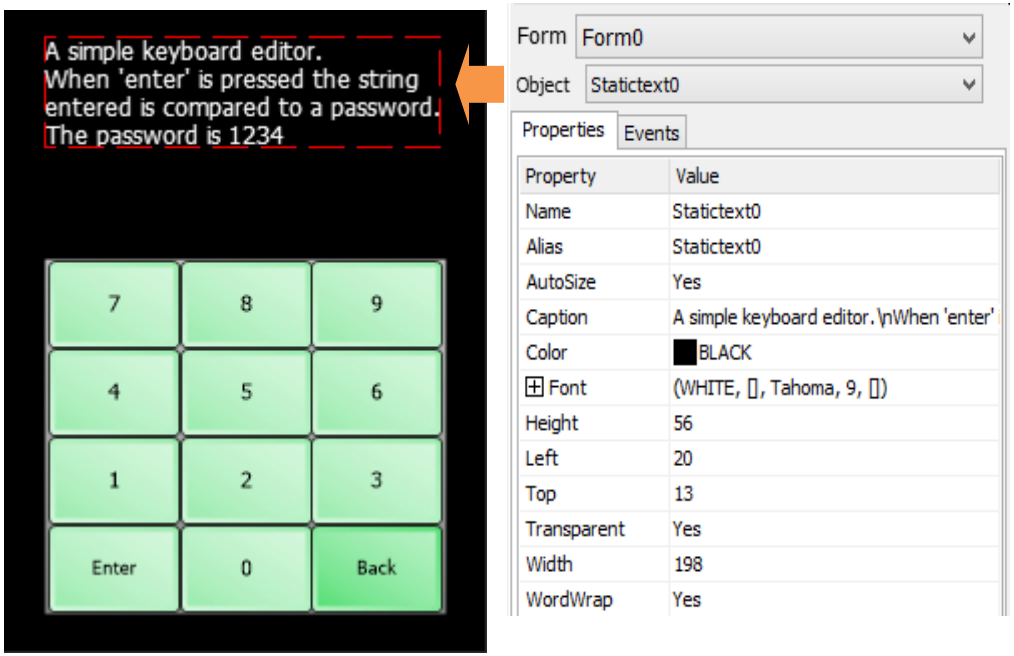
Design the Project

A uLCD-32PTU (portrait orientation) will be used for this application note.

Add a Static Text to Form 0

The static text object under the Labels pane can be used to create labels.



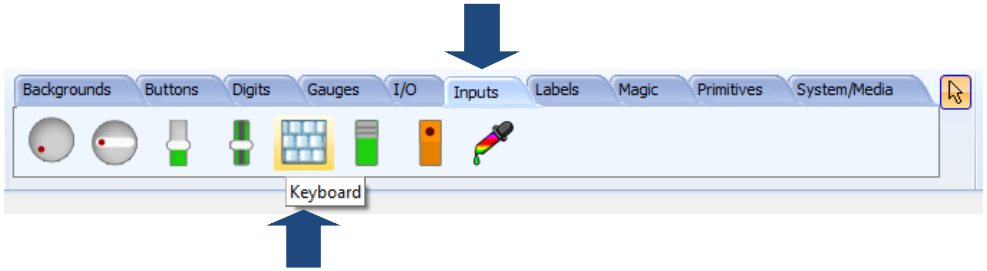


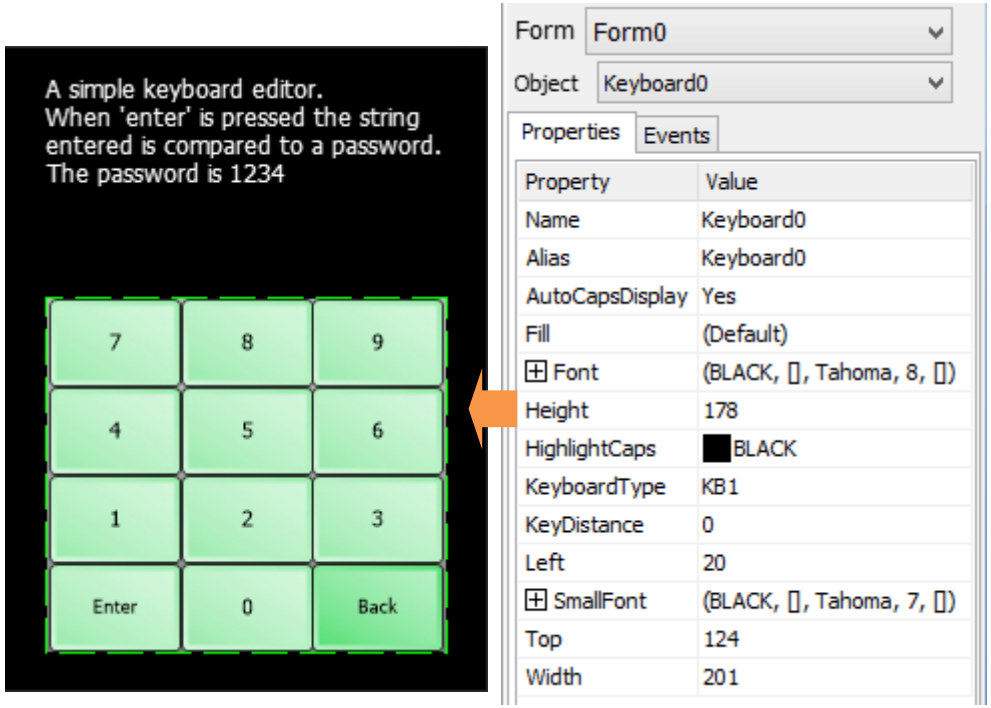
Click on the WYSIWYG screen to place the static text object and edit the properties according to the image shown above.

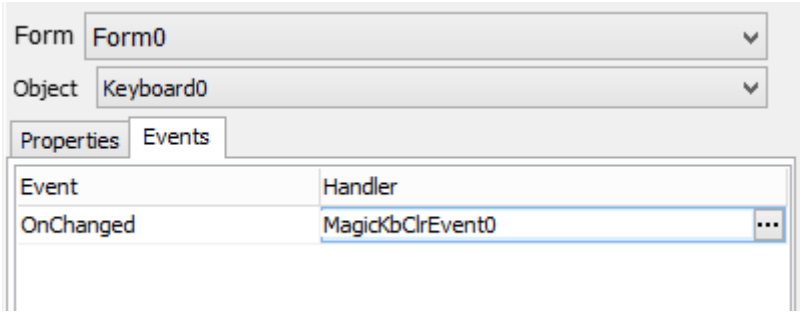
To know more about Static Texts, their properties, and how they are added to a projects, refer to the application note [ViSi-Genie Labels, Texts, and Strings].

Add a Keyboard to Form 0

The keyboard object under the Inputs pane can be used for different applications that needs QWERTY, NUMERIC, CELLPHONE or CUSTOM input.





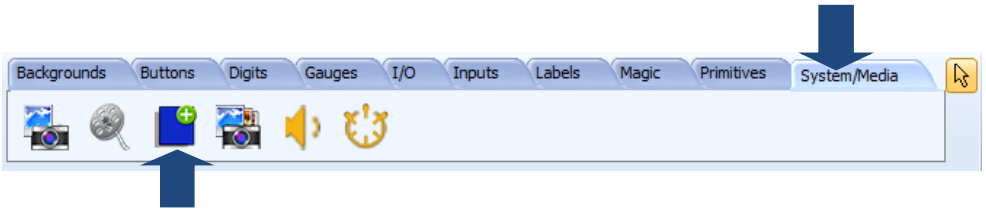


The OnChanged event of the keyboard object is set to MagicKbClrEvent0 so that the pressed keys from the keyboard will be received by the MagicKbClrEvent0.

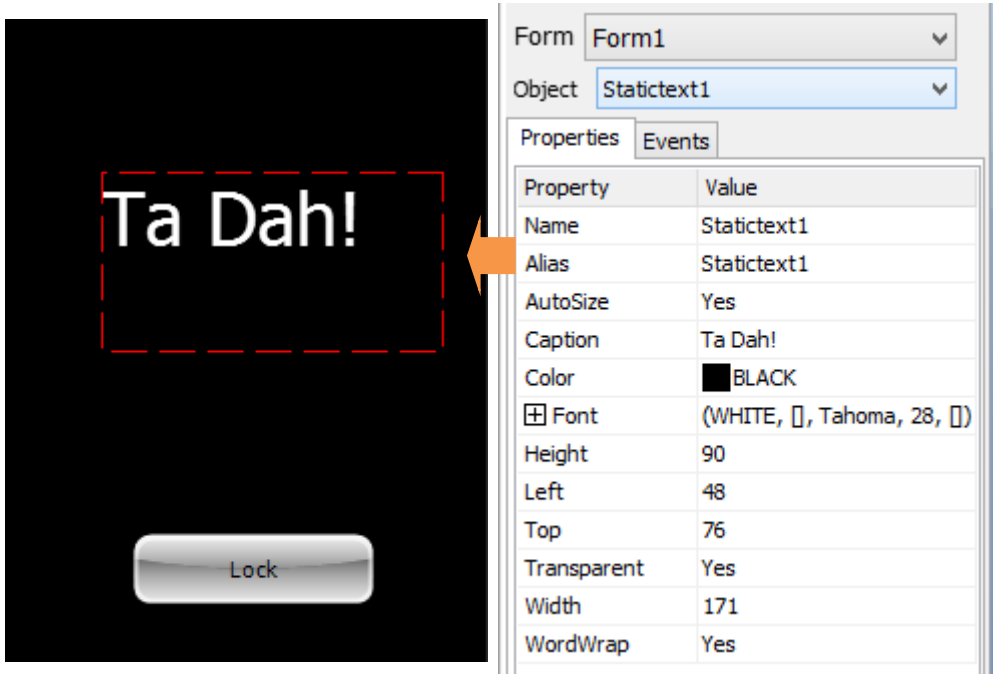
To know more about customizing a Keyboard Object, refer to the application note ViSi-Genie Customised Keyboard.

Create a new Form

In System/Media pane, click the Form object icon shown below.



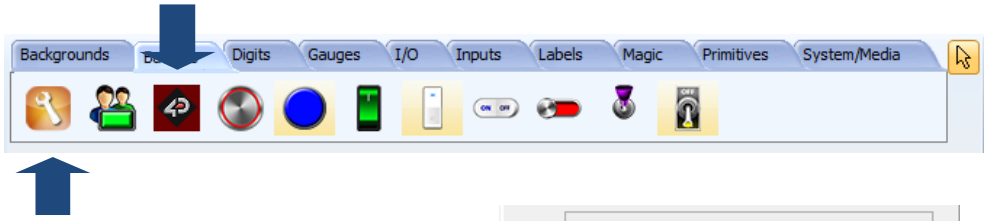
Add a Static Text to Form 1

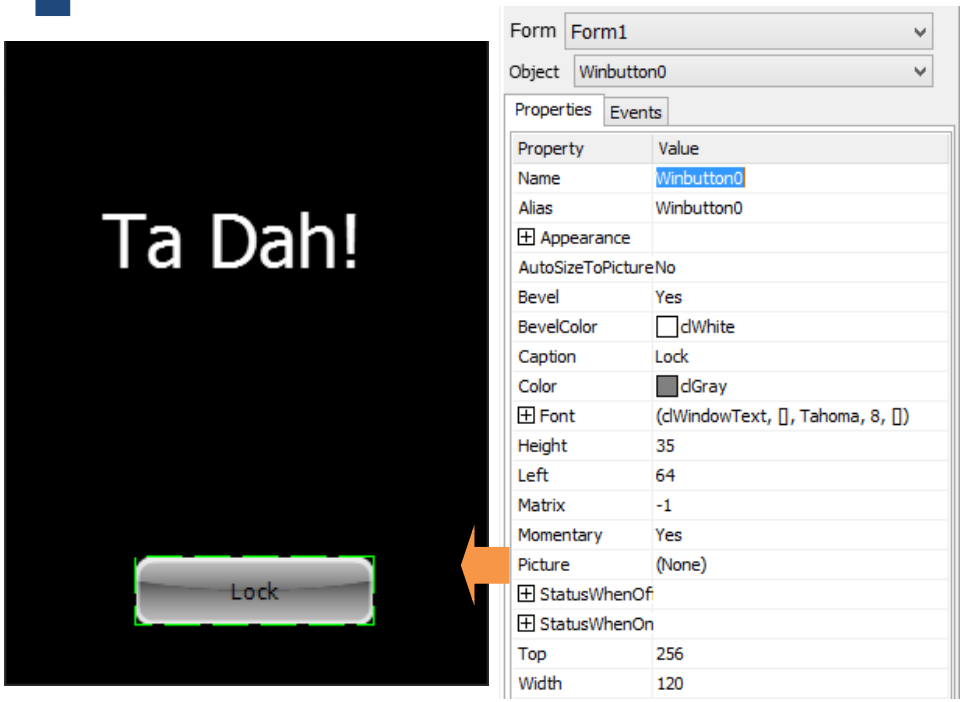


Click on the WYSIWYG screen to place the static text object and edit the properties according to the image shown above.

Add a Fancy Button to Form 1

In Buttons pane, click the Fancy Button icon shown below.





The OnChanged event of Winbutton0 is set to 'Form0Activate' to be able to return to Fom0 if the button is pressed.

To know more about Fancy buttons, refer to the application note: ViSi-Genie Customised Keyboard.

Add Magic Code

To know more about adding Magic Objects, refer to the application note ViSi-Genie How to Add Magic Objects.

MagicCode1

```
1   //
2   // Added 3/11/2014 2:46:14 PM
3   //
4   // MagicCode1
5   //
6
7   func resetchars()
8       txt_MoveCursor(6,lft);
9       print("      ") ;
10      txt_MoveCursor(6,lft);
11  endfunc
12
13  #constant maxresp 4
14  #constant lft     10
15  //
16  var respchp, chars ;
17  var Response[3] ;
```

The image shown is the 4DGL code inside of the Magic Code object. MagicCode1 is the declaration of variables, constant and resetchars() function.

The 'maxresp' constant is the maximum number of input for the password. The 'lft' constant is the start position for echo of *s. And the resetchars function is a timer routine used to blank 'Failed' from the screen.

MagicCode2

```
// init variables used on form0
    if (CurrentForm == 0)
        respchp := str_Ptr(Response) ;
        chars := 0 ;
    endif
```

The image above shows the content of MagicCode2. This magic code's insert point is set to 'PostActivateForm'. This checks if current form is Form0 and then set keyboard response to start.

Add a Keyboard + ColorPicker Event Magic

To know more about adding Keyboard + ColorPicker Event Magic, refer to the application note ViSi-Genie How to Add Magic Objects.

MagicKbClrEvent0

This magic object receives the value inputted on the keyboard object. The value is stored in 'var value'. If there are multiple keyboard objects in a project, then their 'OnChanged' event should be set to the 'Keyboard + ColorPicker Event Magic' that will handle the input value.

```
func rMagicKbClrEvent0(var reportID, var objType, var objHash, var value)
    var i ;
    if (value == 8) // 'back'
        if (chars)
            respchp-- ;
            chars-- ;
            txt_MoveCursor(6,lft+chars);
            putch(' ') ;
            txt_MoveCursor(6,lft+chars);
        endif
```

The image shown above is a part of the content of MagicKbClrEvent0. This part of the program checks if the input value from Keyboard0 is 'Back'. The character 'Back' of keyboard has a decimal value '8'. This simply checks if a character is already printed and then replaces it with space ' ' after moving the text cursor to the character that will be deleted.

```
else if (value == '*') // 'enter'
    str_PutByte(respchp++, 0) ;
    i := str_Ptr(Response) ;
    txt_MoveCursor(6,lft);
    if (str_Match(&i, "1234"))
          print("Match") ;
        ActivateForm(1) ;
    else
        txt_FGcolour(RED);              // 0  t
        print("Fail") ;
        txt_FGcolour(LIME);              // 0
        sys_SetTimerEvent(TIMER0, resetchars);
        sys_SetTimer(TIMER0,500) ;
    endif
    respchp := str_Ptr(Response) ;
    chars := 0 ;
```

Next, the image shown above is the part of MagicKbClrEvent0.inc that will execute if 'Enter' is pressed on the keyboard. This simply uses str_Match to compare the content of the array 'Response' to "1234". If it does match then it will proceed in activating FORM1. Else if the content of 'Response' array does not match "1234" then it will print "Fail" then perform the resetchars() timer routine.

```
else if (chars < maxresp)
    str_PutByte(respchp++, value) ;
    txt_MoveCursor(6,lft+chars);
    chars++ ;
      putch(value) ;
    putch('*') ;
endif
```

Lastly the image shown above is part of MagicKbClrEvent0.inc that will execute if numbers 0-9 is pressed on the keyboard object. This increases the column of the cursor and then prints asterisk '*'. if you want to print the value then uncomment "putch(value)" then comment out "putch('*')".

To know about 4DGL programming please refer to 4DGL Programmer's Reference Manual and GOLDELOX, PICASO, DIABLO Internal Functions Manual.

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.

Proprietary Information

Disclaimer of Warranties & Limitation of Liability