APPLICATION NOTES

4D SYSTEMS

TURNING TECHNOLOGY INTO ART

ViSi Displaying Images from the uSD Card – WYSIWYG RAW

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

WWW.4DSYSTEMS.COM.AU

Description

This application note is dedicated to illustrating how an image is displayed on a Goldelox module using the ViSi environment in Worskhop 4. ViSi allows the user to place a widget (an image object for example) on the WYSIWYG (What-You-See-is-What-You-Get) screen and paste the corresponding 4DGL code to the code area. Worksop 4 then automatically generates the graphics files in the background and copies them to the uSD card. The 4DGL code is compiled and downloaded to the Goldelox display module. When the program runs, it accesses the graphics files on the uSD card and displays the desired object on the screen.

In order to carry out this application, the following items are required:

- Any of the following 4D Goldelox display modules:

  uOLED-96-G2
  uOLED-128-G2
  uOLED-160-G2
  uLCD-144-G2
  uTOLED-20-G2

  or any superseded module that supports the ViSi environment
- 4D Programming Cable or µUSB-PA5
- 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

Displaying an image on a 4D screen is one of the most quintessential applications. This application note will walk through the steps involved in decoding an image into 4D format; the procedure required to place this onto an external µSD card; the 4DGL code required for displaying the image; and a brief insight into the various image control functions available that can manipulate the visual appearance.

Setup Procedure

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

[ViSi Getting Started - First Project for Goldelox]

Create a New Project

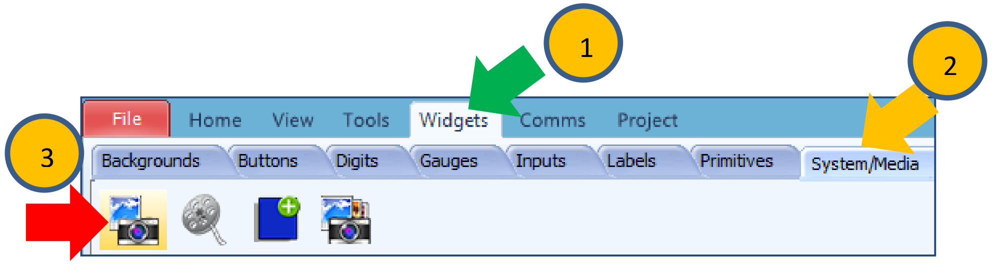
For instructions on how to create a new **ViSi** project, please refer to the section "**Create a New Project**" of the application note

[ViSi Getting Started - First Project for Goldelox]

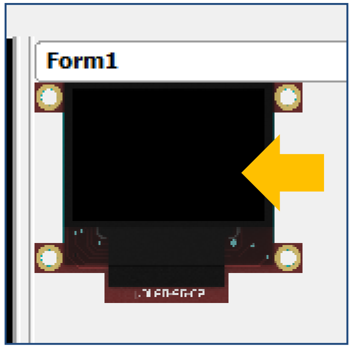
Design the Project

Adding an Image Object

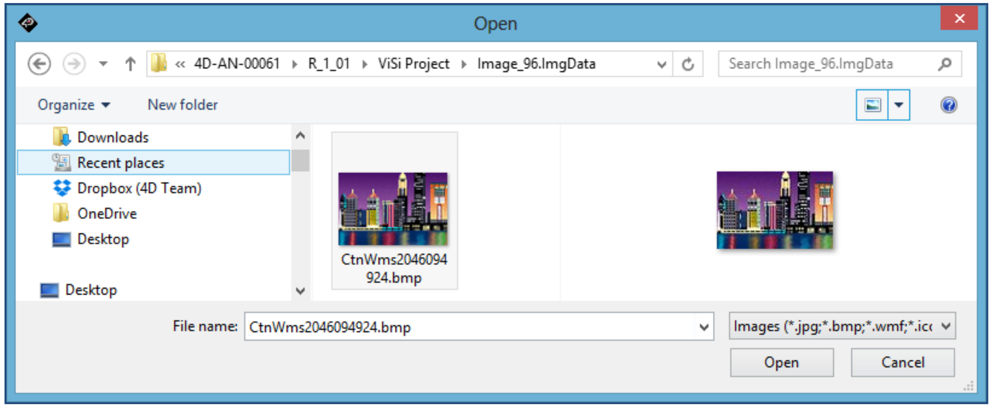
Go to "Widgets" tab in the program window, select the System/Media tab, and select the image widget icon.



Now click on the WYSIWYG screen to place the image object.

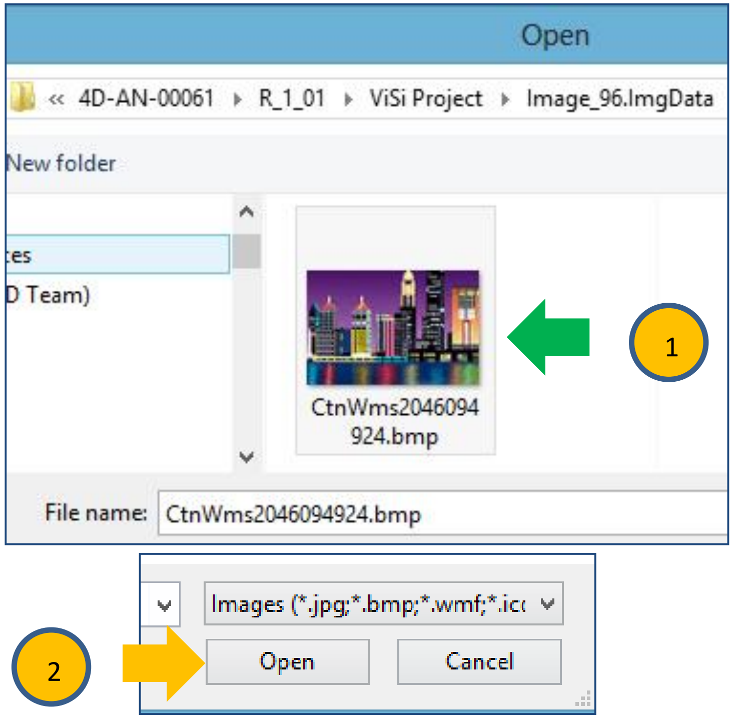


A dialogue box will open up .Browse for and select the image, then click open.



File name: CtnWms2046094924.bmp

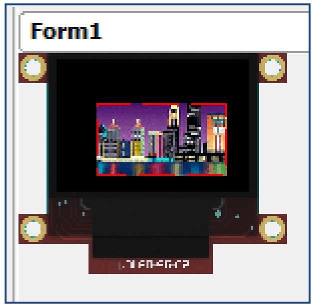
Images (*.jpg;*.bmp;*.wmf;*.ic



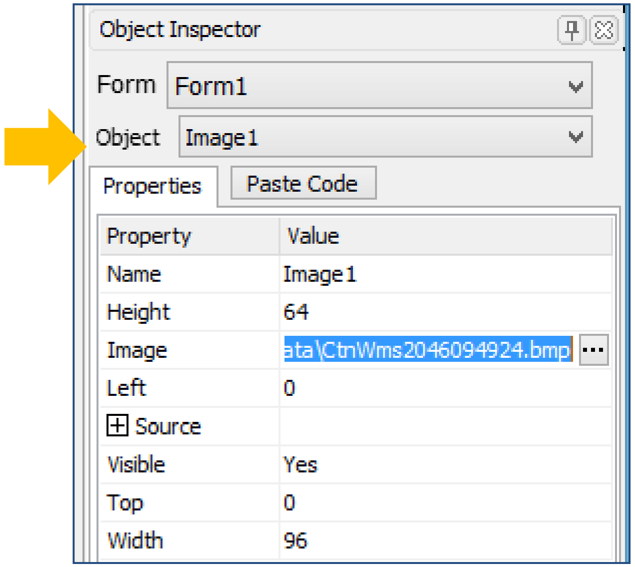
Open
« 4D-AN-00061 ▸ R_1_01 ▸ ViSi Project ▸ Image_96.ImgData
New folder
es
D Team)
CtnWms2046094924.bmp
1
2
Images (*.jpg;*.bmp;*.wmf;*.ic
Open
Cancel

File name: CtnWms2046094924.bmp

Now, adjust the properties of the image to your requirements by dragging on the read border lines.



The object inspector lists the properties of the newly added image object. The values of the properties can be set using the object inspector. The object has the name "Image1".



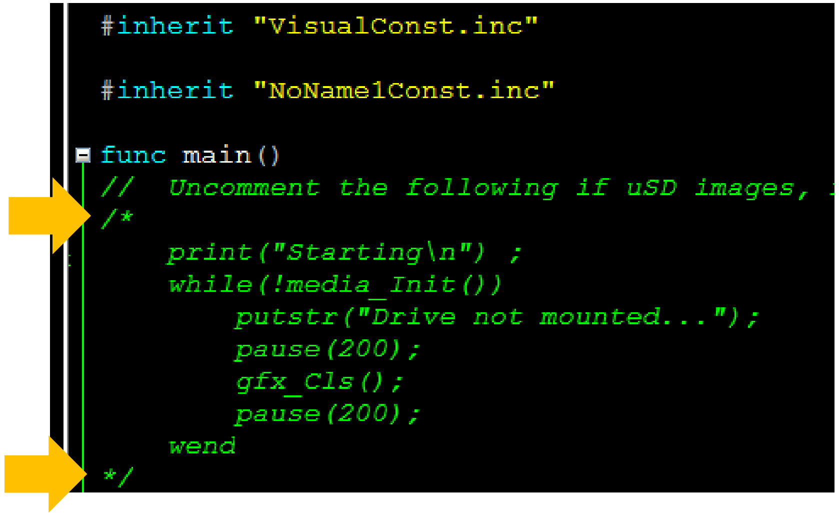
Form: Form1

Object: Image1

Property	Value
Name	Image1
Height	64
Image	ata\CtnWms2046094924.bmp
Left	0
⊞ Source	
Visible	Yes
Top	0
Width	96

uSD Card Initialization Routine

For this project a uSD card will be needed. Before its contents can be accessed, a uSD card needs to be initialized first. There is a uSD card initialization routine included in the default code of a ViSi project. Uncomment it by removing the block comment symbols, as shown below.

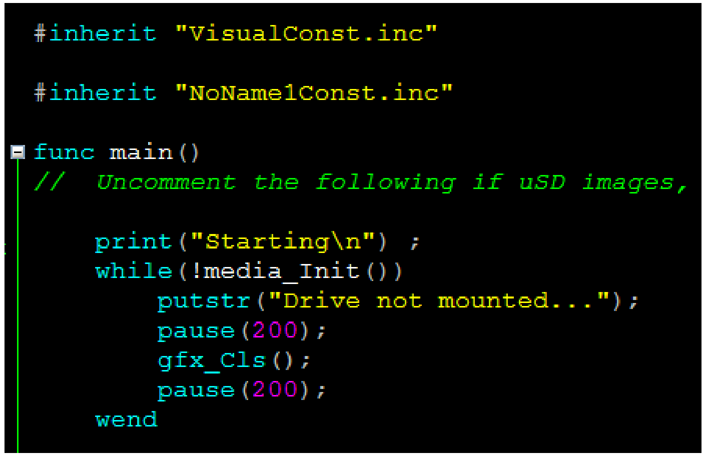


```
#inherit "VisualConst.inc"

#inherit "NoName1Const.inc"

func main()
//  Uncomment the following if uSD images,
/*
    print("Starting\n") ;
    while(!media_Init())
        putstr("Drive not mounted...");
        pause(200);
        gfx_Cls();
        pause(200);
    wend
*/
```

remove

remove

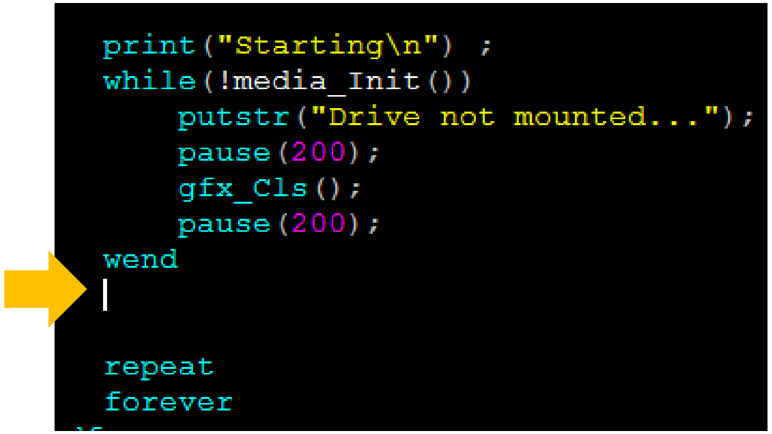
The routine should now be a part of the code.



```
#inherit "VisualConst.inc"

#inherit "NoName1Const.inc"

func main()
//  Uncomment the following if uSD images,

    print("Starting\n") ;
    while(!media_Init())
        putstr("Drive not mounted...");
        pause(200);
        gfx_Cls();
        pause(200);
    wend
```

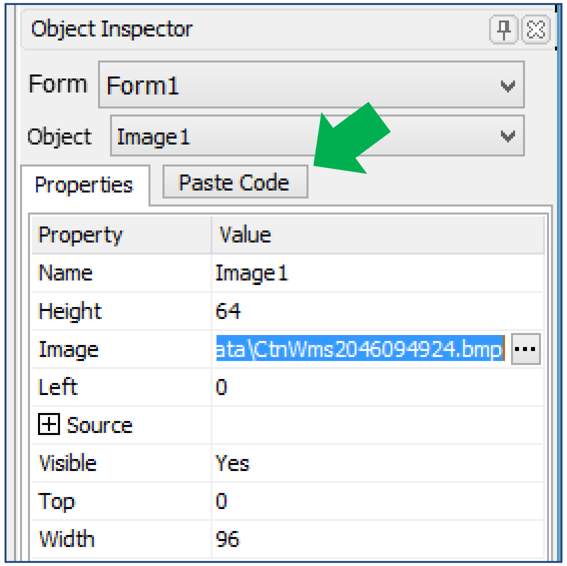
Paste the Code for the Image Object

Now, put your cursor just after the uSD card initialization routine.



```
print("Starting\n") ;
while(!media_Init())
    putstr("Drive not mounted...");
    pause(200);
    gfx_Cls();
    pause(200);
wend
|

repeat
forever
```

Click on the “Paste Code” button of the Object Inspector for the object as shown below.



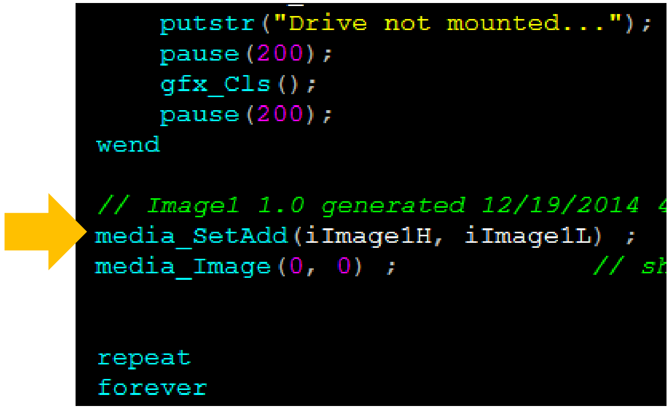
Form: Form1

Object: Image1

Properties

Property	Value
Name	Image1
Height	64
Image	ata\CtnWms2046094924.bmp
Left	0
⊞ Source	
Visible	Yes
Top	0
Width	96

The code area should be updated accordingly.



```
    putstr("Drive not mounted...");
    pause(200);
    gfx_Cls();
    pause(200);
wend

// Image1 1.0 generated 12/19/2014 4
media_SetAdd(iImage1H, iImage1L) ;
media_Image(0, 0) ;              // sh


repeat
forever
```

Run the Program

For instructions on how to save a **ViSi** project, how to connect the target display to the PC, and how to compile and download a program, please refer to the section "**Run the Program**" of the application note

ViSi Getting Started - First Project for Goldelox

Editing the Code

The user can change the x, y location of the image by simply changing the x, y arguments in the following command,

```
media_Image(0, 0) ;          // show image
```

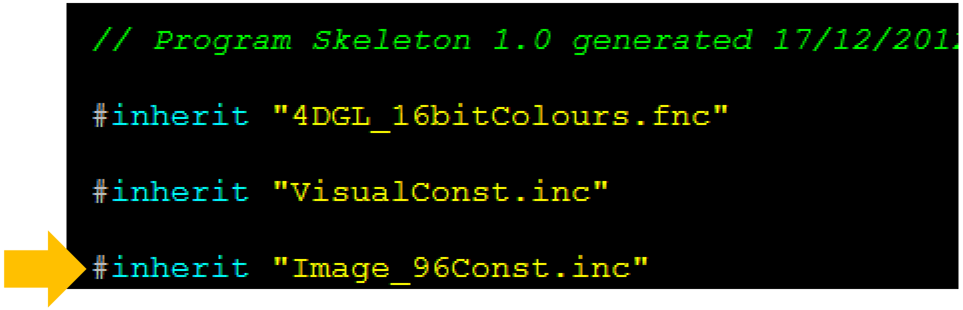
If the user wishes to change the width, height, and other properties or replace the image with another image, he needs to remove the relevant piece of code, which in this case would be,

```
// Image1 1.0 generated 17/12/2012 11:32:49 AM
media_SetAdd(iImage1H, iImage1L) ;     // point to the Image1 image
media_Image(0, 0) ;          // show image
```

Then, change the properties of the image and "Paste Code" again. Please note, once the code is pasted, the changes in the Properties do not reflect on to the code i.e. all the properties need to be set prior to pasting the code.

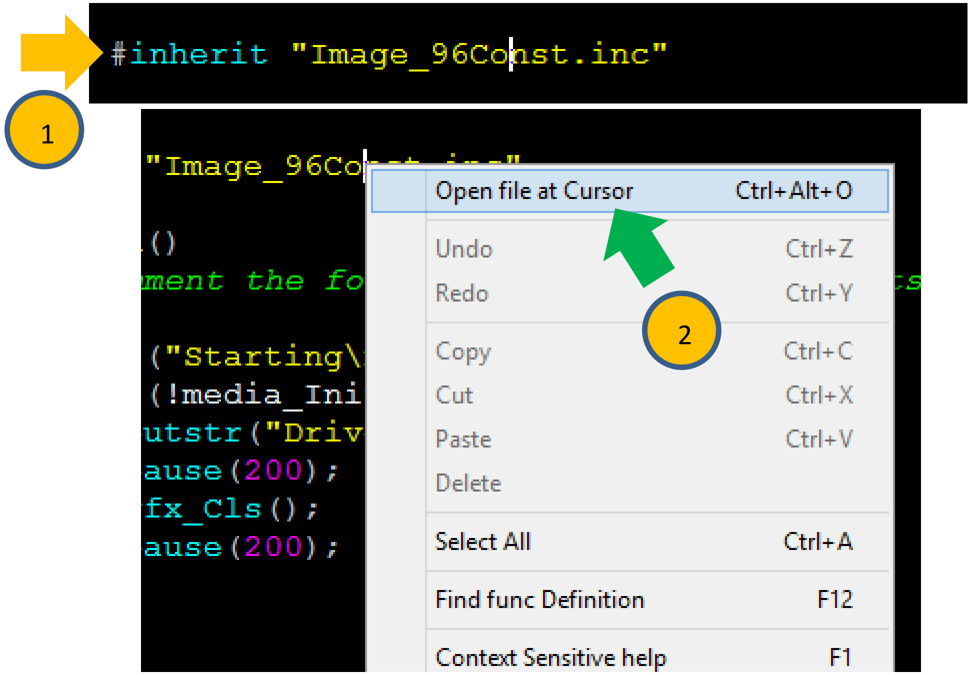
The arguments "**iImage1H**" and "**iImage1L**" of the function "**media_SetAdd()**" are the high word and low word of the address of the location of the object Image1 in the uSD card. These are actually constants the values of which are automatically generated by Worskhop. Before any object can be displayed, its location or address in the uSD cad needs to be specified by using the above function. To know the exact value of the constants, follow the steps below.

1. Save and compile the project.
2. Go to the beginning part of the code where the header files are listed. Workshop automatically generates a header file the name of which is derived from that of the project. For our working project with the filename "Image_96", the header file with which we are interested is "Image_96Const.inc".



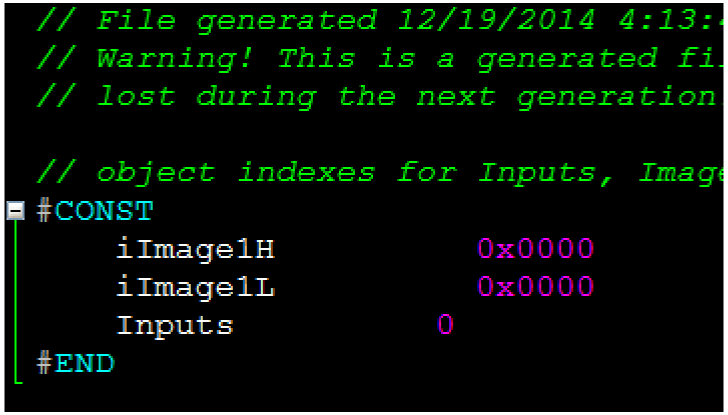
```
// Program Skeleton 1.0 generated 17/12/201

#inherit "4DGL_16bitColours.fnc"

#inherit "VisualConst.inc"

#inherit "Image_96Const.inc"
```

3. Put the cursor anywhere on the filename text and click on the right mouse button. A drop-down menu appears. Select the first option "Open file at Cursor".



```
#inherit "Image_96Const.inc"
```

4. The header file should now open.



```
// File generated 12/19/2014 4:13:
// Warning! This is a generated fi
// lost during the next generation

// object indexes for Inputs, Image
#CONST
    iImage1H            0x0000
    iImage1L            0x0000
    Inputs          0
#END
```

Every time that an object is added to the project and the project is compiled, this header file is regenerated to include the addresses of the new objects

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.