APPLICATION NOTES

4D SYSTEMS
TURNING TECHNOLOGY INTO ART

ViSi-Genie Play Video

DOCUMENT DATE: **25th APRIL 2019**
DOCUMENT REVISION: **1.2**

Description

This Application Note explores the possibilities provided by ViSi-Genie for the **Movie** object:

- Play
- Pause
- Stop
- Change volume

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

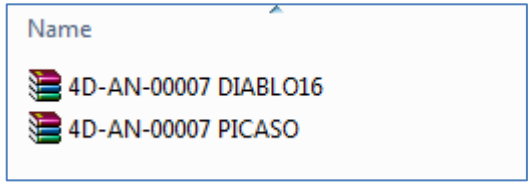
Application Overview

Adding video to a graphical user interface increases the user experience dramatically. 4D Systems screens feature video playing.
The application discussed in this application note is a fully featured music player:

ViSi-Genie makes building such an application as simple as click-and-drop elements on the screen.
This application note describes how to add a **Video** object and how to customise it.

Setup Procedure

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



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
ViSi Genie Getting Started – First Project for Diablo16 Displays

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

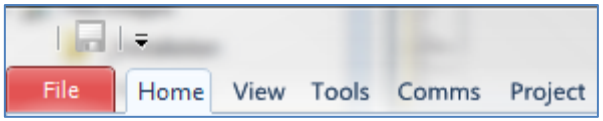
ViSi Genie Getting Started – First Project for Picaso Displays
ViSi Genie Getting Started – First Project for Diablo16 Displays

The Video Object

You can load the example…

Example: 4D-AN-00007 PICASO – Play Video 2 or 4D-AN-00007 DIABLO16 – Play Video 2

…or follow the procedures described hereafter.
Select the **Home** menu to display the objects:



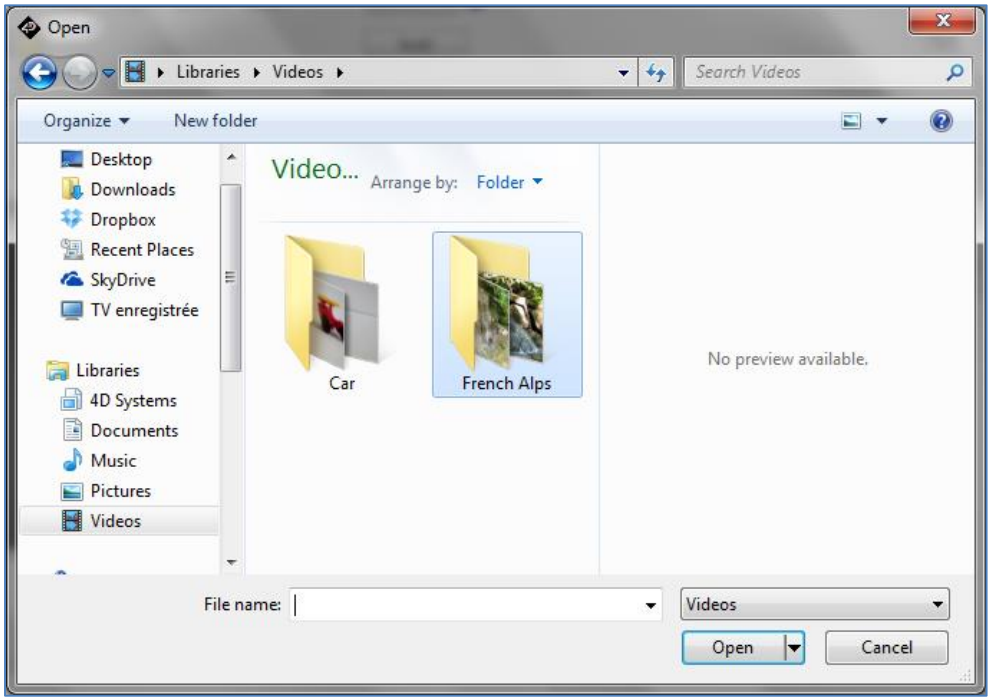
The **Video** object is located on the System/Media pane:

Add a Video Object

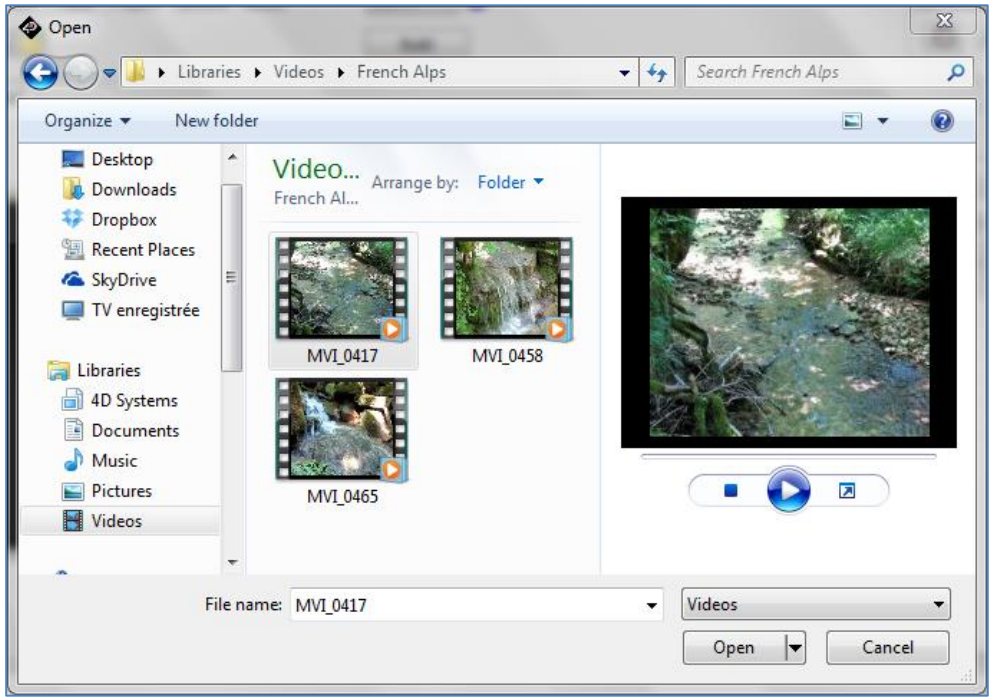
Click first on the video icon…

…and click on the WYSIWYG screen to place it.

The standard Windows **Open** file appears and asks for a video:



Select the folder, here *French Alps*, and the picture, here *MVI_0471* with a quiet creek in the Alps…

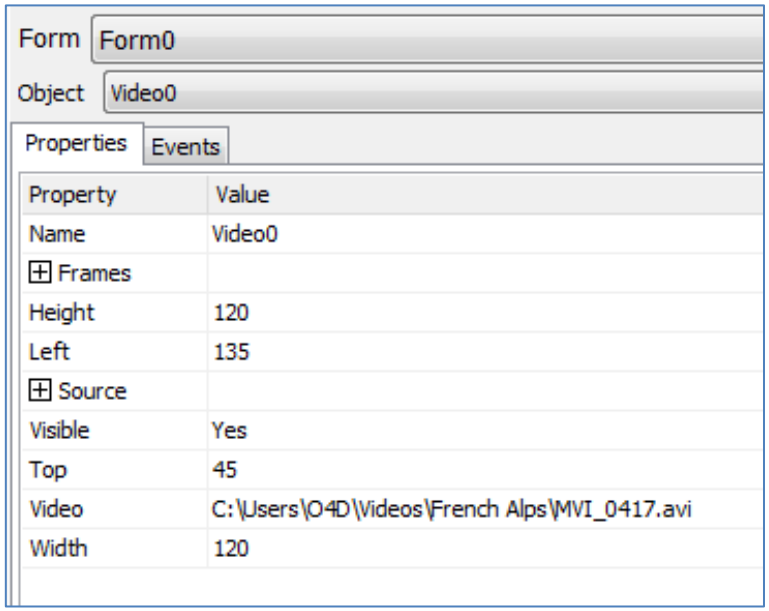


...and press **Open** to load the video:

The WYSIWYG screen now displays the video:

Note that the video is in a square in the middle of the screen and that the proportions are not consistent with the original ones.

The Object Inspector displays the properties of the video object:



Form: Form0

Object: Video0

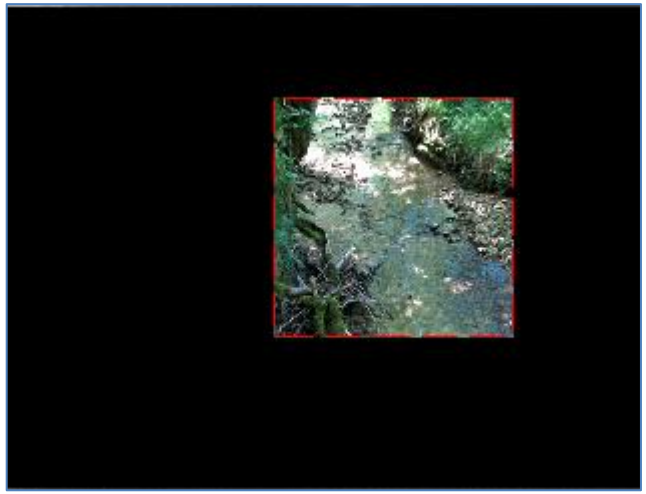
Properties | Events

Property	Value
Name	Video0
⊞ Frames	
Height	120
Left	135
⊞ Source	
Visible	Yes
Top	45
Video	C:\Users\O4D\Videos\French Alps\MVI_0417.avi
Width	120

The video object contains one single video.

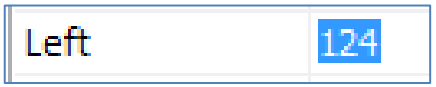
Move the Video

To place the video on the top-right corner of the screen, first select the video object by clicking on it; red dots appear around the video:



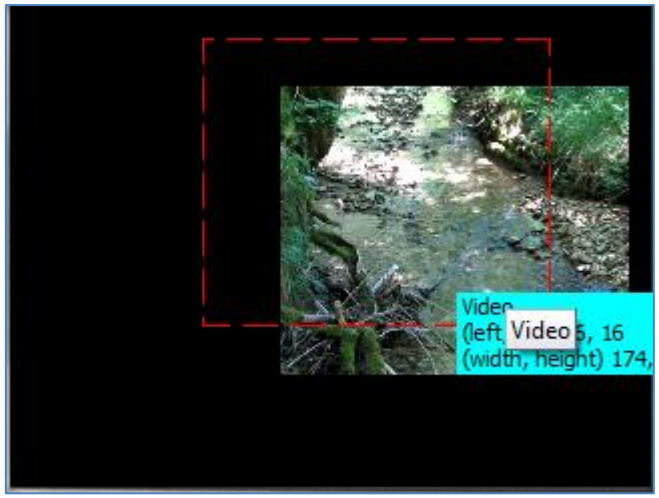
Then, two options:

- Either use the keyboard and edit two properties on the Object Inspector, **Left** and **Top** :
  - Select the **Left** line, enter *0* and press **Enter**



  - Select the **Top** line, enter *0* and press **Enter**

- Or use the mouse, click-and-drag the video on the top-left corner of the screen:



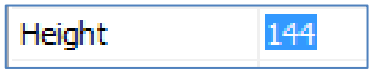
The WYSIWYG screen shows the new location of the video:

Resize the Video

To resize the video, first select the video object by clicking on it; red dots appear around the video:

Then, two options:

- Either use the keyboard and edit two properties on the Object Inspector, **Height** and **Width** :
  - Select the **Height** line, enter *0* and press **Enter**



  - Select the **Width** line, enter *0* and press **Enter**

- Or use the mouse, click on the bottom-left red dots and resize the video to the desired dimension:

The WYSIWYG screen shows the new size of the video:

The video can also be full-sized:

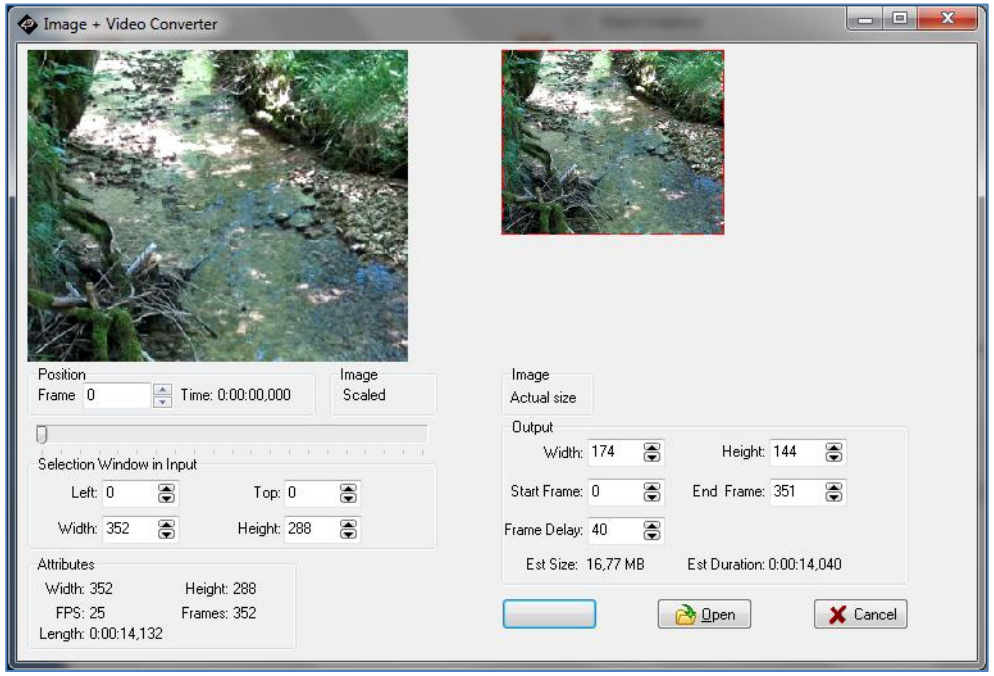
Because the screen used for this application note is 320 x 240, maximal height is 240 pixels and maximal width is 320 pixels. Those values may change depending on your specific screen. Please refer to the specification sheet of your screen.

Edit the Video

To edit the video, click on the video line in the **Object Inspector**:

A  symbol appears. Click on it.

A new window **Image – Video Converter** appears and provides all the parameters for the video:



On the left side, the input video; on the right, the output video.

Change the Video

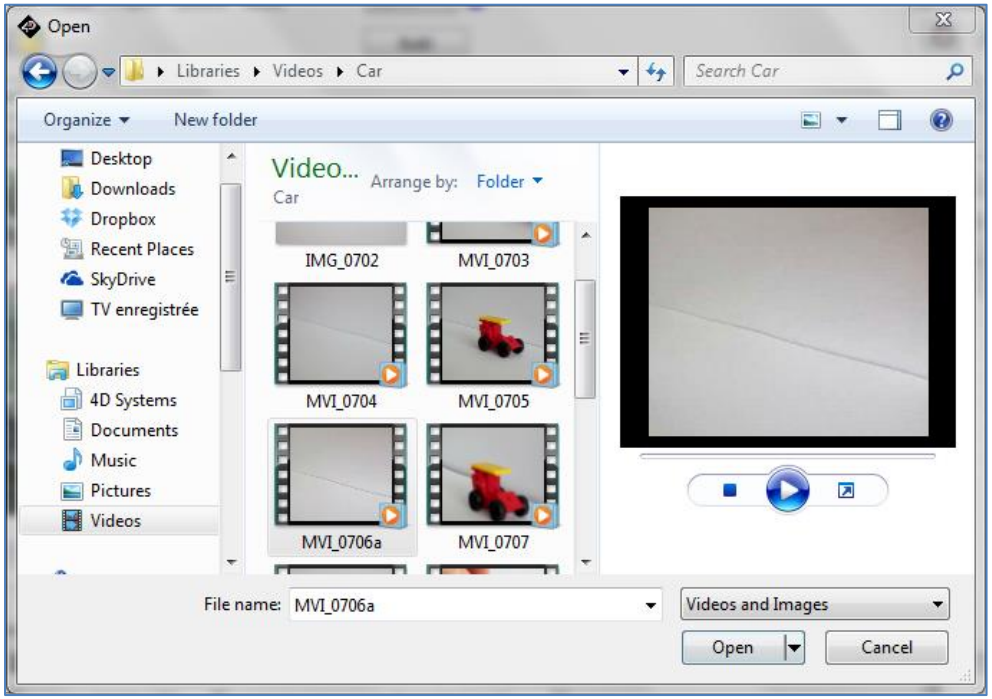
You can load the example…

Example: 4D-AN-00007 PICASO – Play Video 3 or 4D-AN-00007 DIABLO16 – Play Video 3

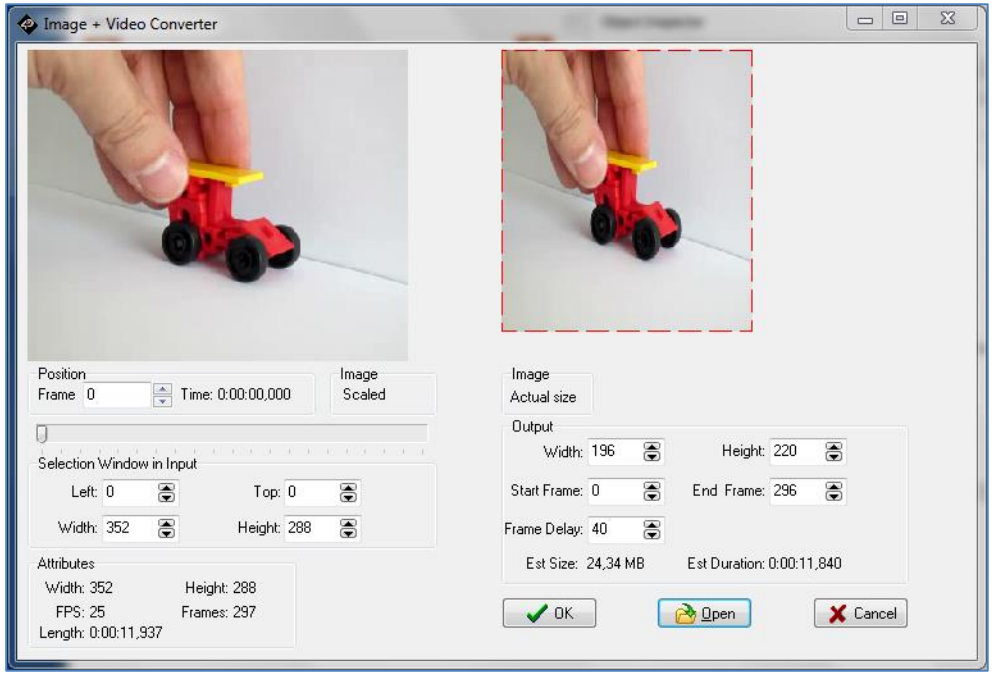
…or follow the procedures described hereafter.

To load a new video, click on

The standard Windows **Open** file appears and asks for a video:



The **Image – Video Converter** shows the new video:

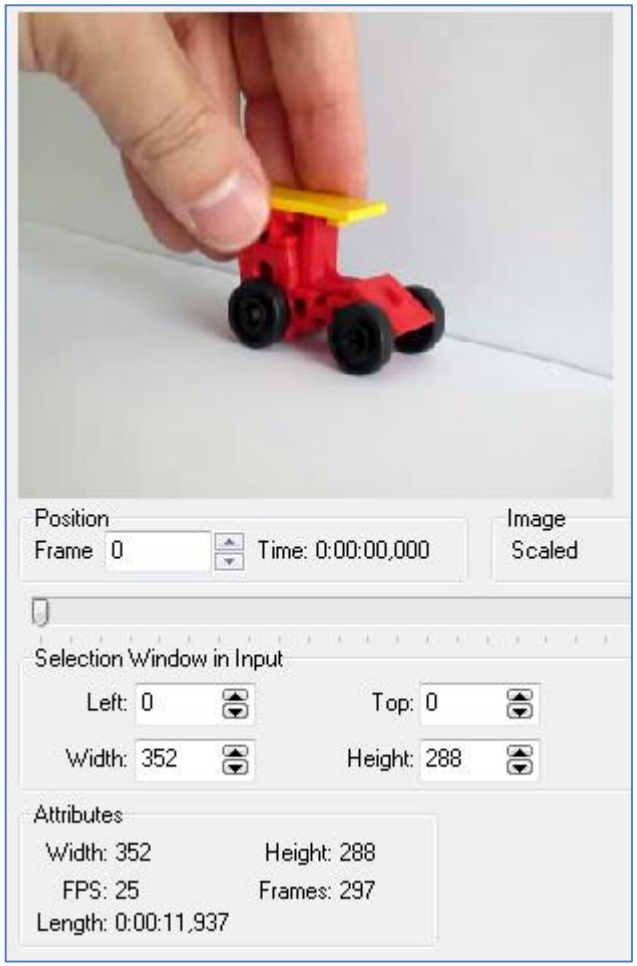


Note that all the settings have been changed.

This new video shows a toy car rolling from left to right.

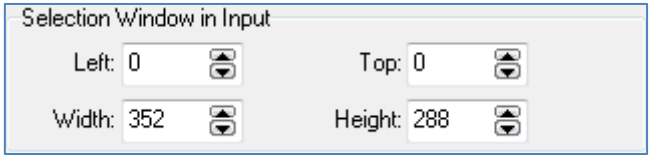
Crop the Input Video

The left side provides all the information about the input video:



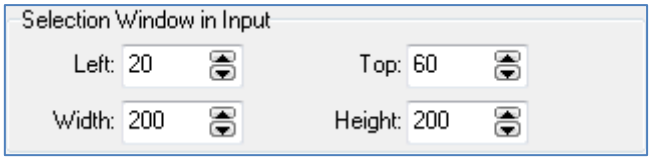
Cropping part the input image is possible by

- Entering new values for **Left**, **Top**, **Width** and **Height** under **Selection Window in Input**:

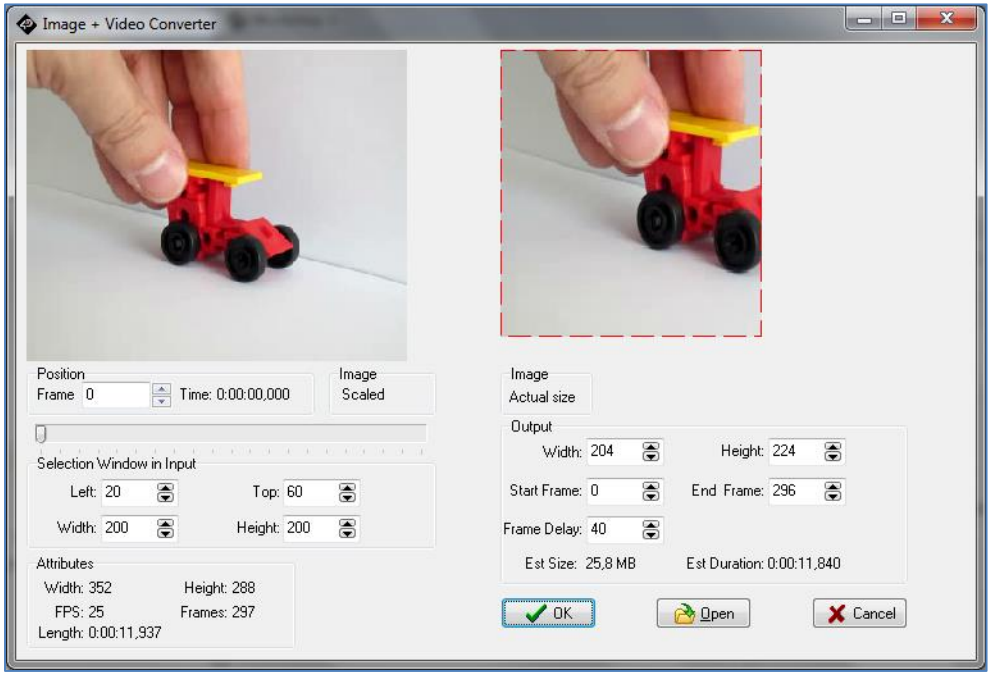


- Clicking on the up or down arrows for the concerned value:

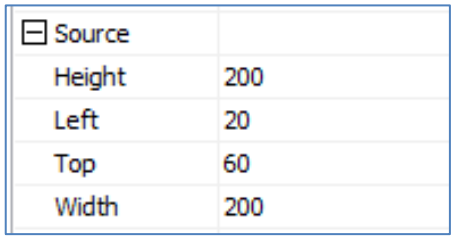
Focusing on the car gives the following result:



Only the image on the right is updated:



Cropping the input image can also be done by changing the **Height**, **Left**, **Top** and **Width** properties under Source on the **Object Inspector**:

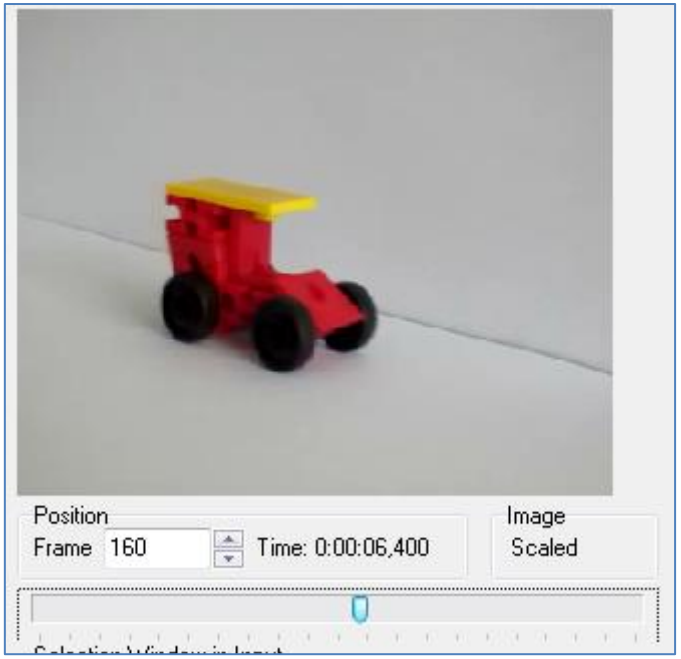


Any format is possible, here an unusual square.

Select a Sequence from the Video

Below the input video, there is a slider to select the frame:

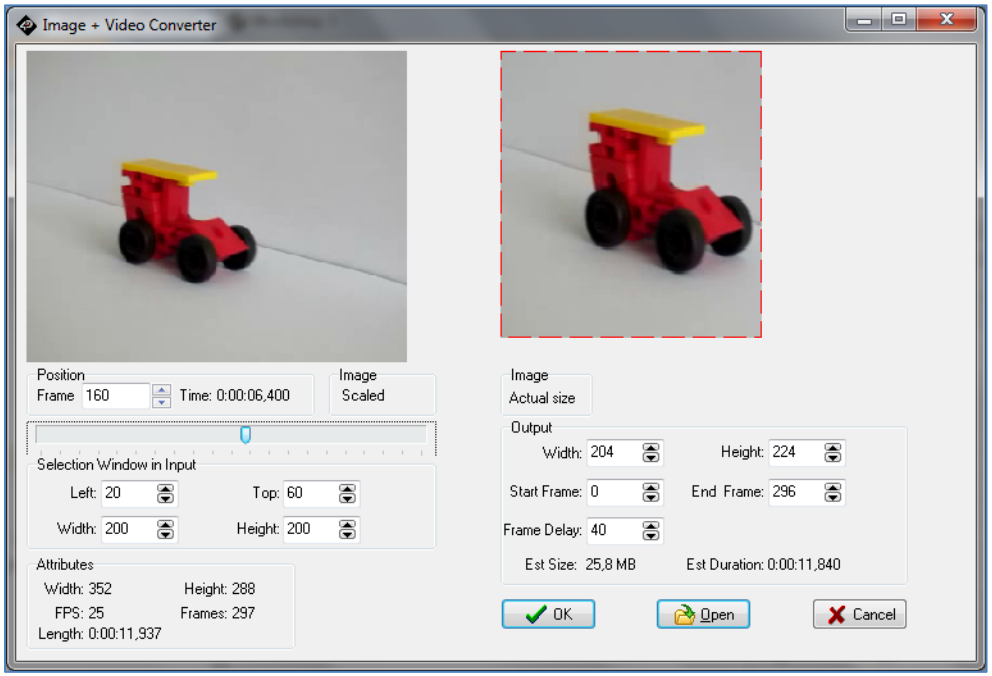
Here, frame 160 is selected and shown:



On the example, the interested sequence on the video starts at frame 100 and ends at frame 220.

There's no need to call for an external editor: just note the numbers of the first and last frames.

The image on the right is updated accordingly:

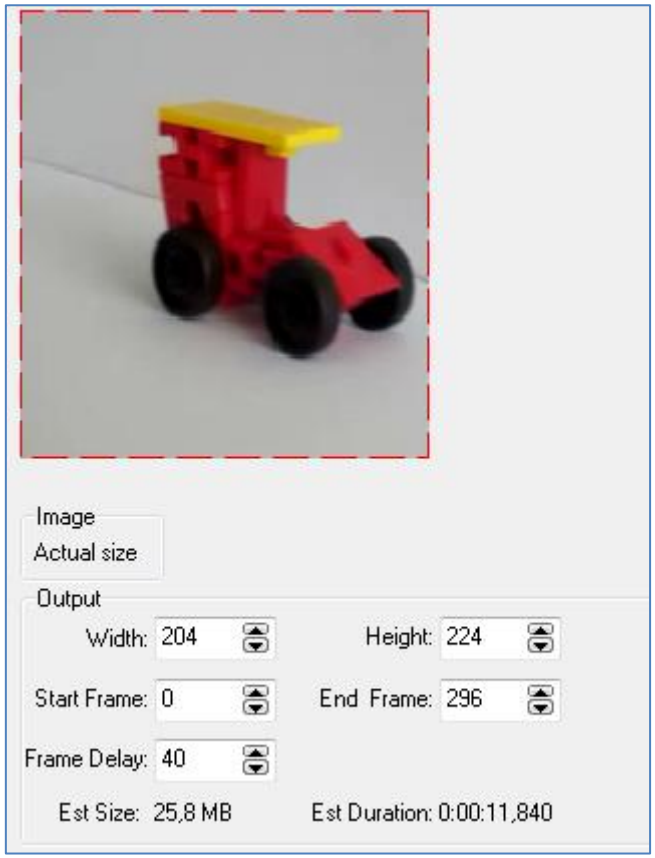


The slider provides a convenient way to review the input video for selecting the sequence.

Keep the numbers of the first and last frames of the sequence for later use.

Resize the Output Image

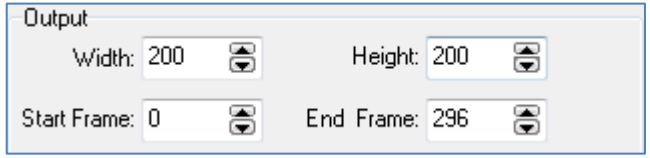
Similarly, the right side displays the output video:



Because the screen used for this application note is 320 x 240, maximal height is 240 pixels and maximal width is 320 pixels. Those values may change depending on your specific screen. Please refer to the specification sheet of your screen.

Changing the size of the output image is possible by:

- Resizing the video moving the red dotted rectangle,
- Entering new values for **Width** and **Height** under **Image | Actual Size | Output**:



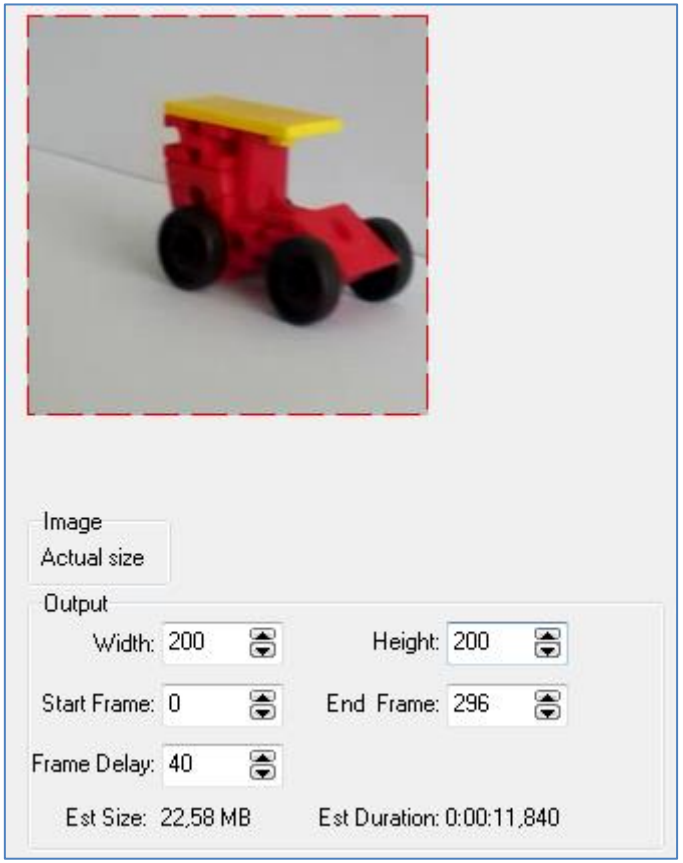
- Clicking on the up or down arrows for the concerned value:

Note it is not possible to move the output image on the screen.

The top-left position of the video is defined by the **Top** and **Left** properties on the **Object Inspector**. Please refer to the **Error! Reference source not found.** section.

It is important to keep the ratio of the output video consistent with the input. The ratio corresponds to width / height. Is this example, input and output share the same width and height, at 200.

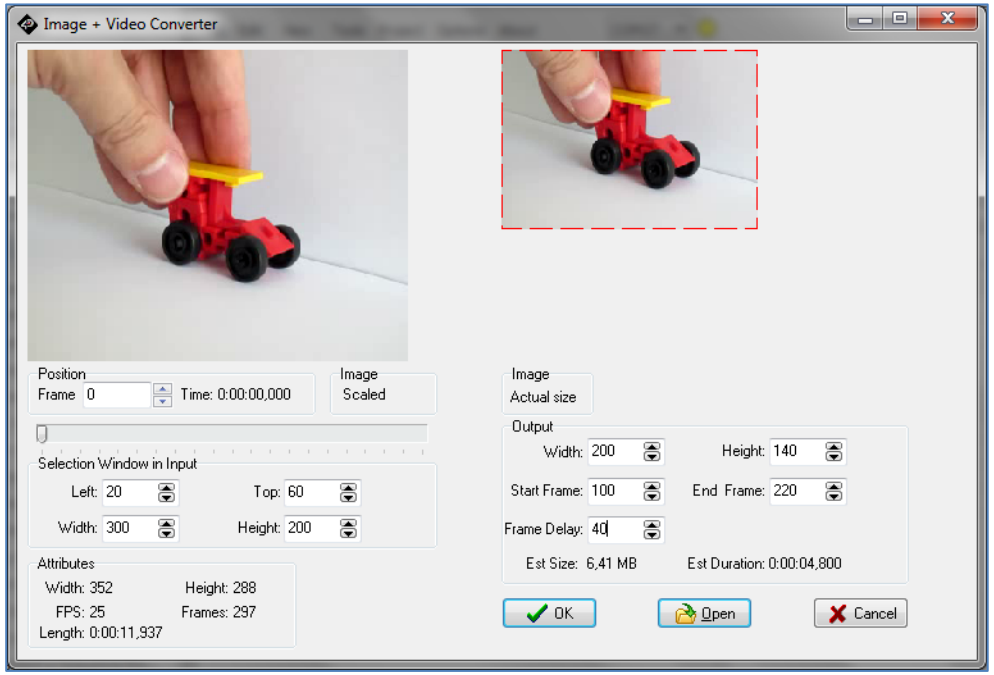
The output image is resized with an estimated size:



The resulting screen would be:

However, most usual ratios are: 3/2, 4/3, 16/9.

Here, the example is updated with a standard 3/2-based screen:



Define the Frame Delay of the Video

Last parameter, the frame delay defines how fast the frames are displayed. Normal speeds are between 25 and 30 images per second.

A speed of 25 images per second corresponds to a frame delay of 40 ms. A speed of 30 images per second corresponds to a frame delay of 30 1/3 ms. The higher the speed, the smaller the delay.

The frame delay has a direct impact on the size of the file: the shorter the delay, the bigger the file because more frames are required.

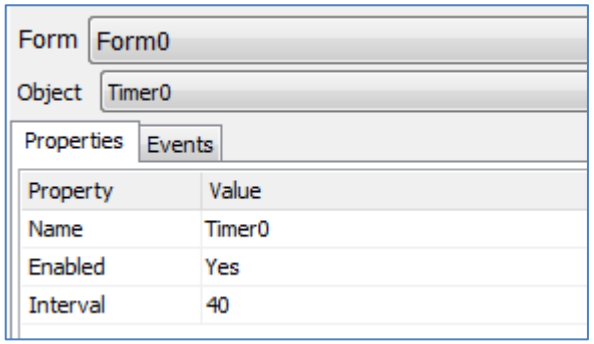
It may also have an impact on the quality of the rendering, as more frames require more power, especially in full screen mode. The screen used for this application note is 320 x 240, maximal height is 240 pixels and maximal width is 320 pixels, or a total of 76 800 pixels. Those values may change depending on your specific screen. Please refer to the specification sheet of your screen.

An example of a full screen video is provided with

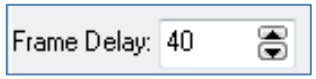
Example: 4D-AN-00007 PICASO – Full Screen or 4D-AN-00007 DIABLO16 – Full Screen

It is important that both the timer and the video have the same value:

- **Interval** for the timer



- And **Frame Delay** for the video:



Confirm or Discard the Changes

When cropping, resizing are finished, click…

…to accept or…

…to discard the changes.

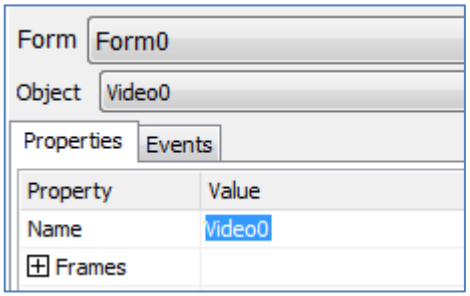
Click on **OK** to accept the new image and settings.

The WYSIWYG screen now shows the car:

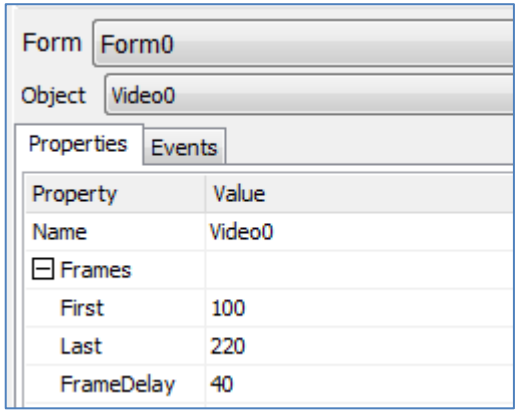
Define the Sequence to Be Played

Using the slider on the **Image – Video Converter** window, the interested sequence on the video starts at frame 100 and ends at frame 220.

To define the sequence, select the **Frames** on the Object Inspector…



…display the details and update **First** and **Last**:



The video is now ready.

Control the Video Object

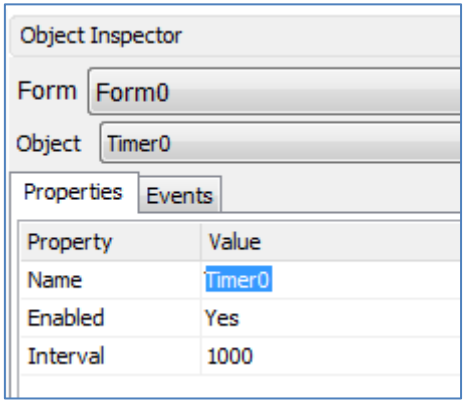
Building and uploading the project as it is now carry a surprise: the video doesn't play!

Add the Timer Object

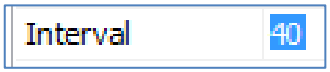
Actually, another object is needed to play video: the **Timer** object…

…located on the System/Media pane

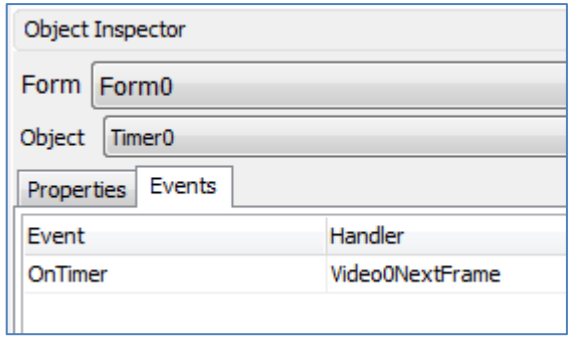
The **Timer** object raises an event at a given pace, for example every 1000 ms:



The **Interval** can be changed easily, by editing the corresponding line, here with 40 ms:



Every 40 ms, **Timer0** raises the event called **OnTimer**.
Now, the **OnTimer** event is defined to send the **Video0NextFrame** command to the Video0 object:



The command **Video0NextFrame** stands for *Tell the **Video0** object to show the next frame of the video.*

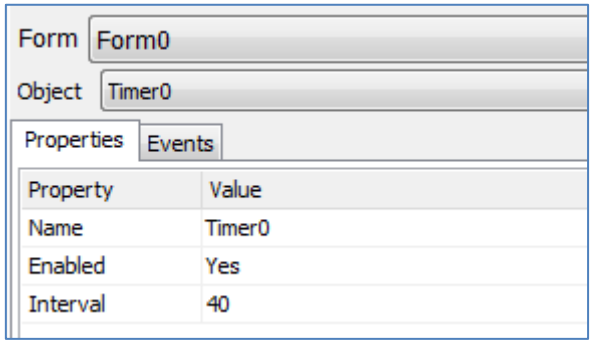
Every 40 ms, **Timer0** raises **OnTimer** event…

…sends **Video0NextFrame** command to…

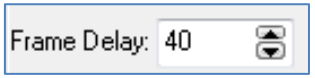
…**Video0** to show the video.

So, every 40 ms, a new frame is going to be displayed. As a video is a succession of still images, changing the images quickly enough creates the impression of a movie.
It is important that both the timer and the video have the same value:

- **Interval** for the timer



- And **Frame Delay** for the video:



Prepare the Buttons

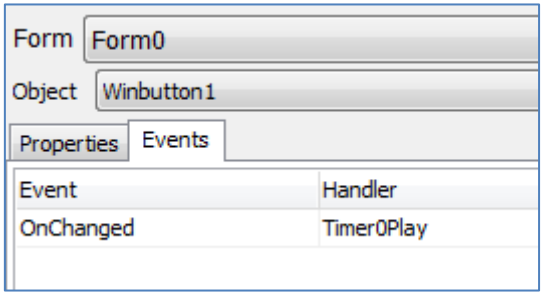
Add five **Button**, a **TrackBar,** a **Label** and a **CustomDigits** objects to the form, customise their appearance properties as you like, in order to obtain the following screen:

Each objet, among the buttons and the track-bar, when it is pressed and released, sends a specific command, directly or indirectly, to control the Video object.

Play the Video

To start playing the video, the **WinButton1** button…

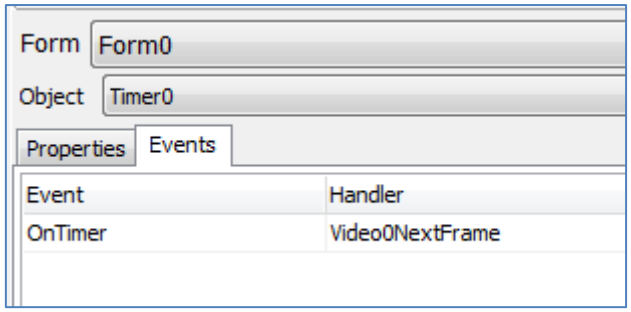
…sends the command **Timer0Play** to **Timer0**:



The command **Timer0Play** stands for *Tell the **Timer0** object to start the timer and raise an event every defined delay.*

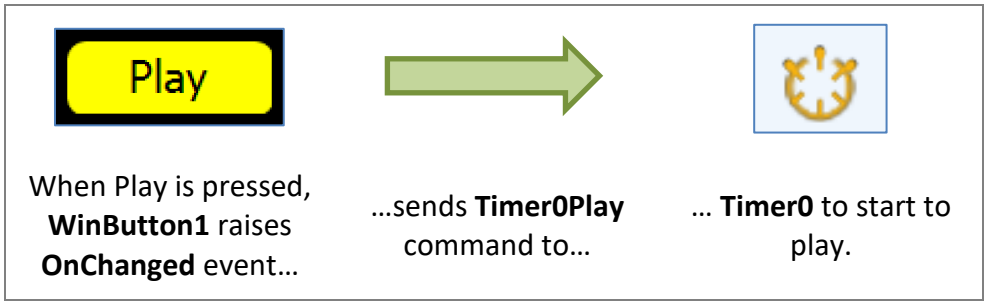
The **Timer0** timer…

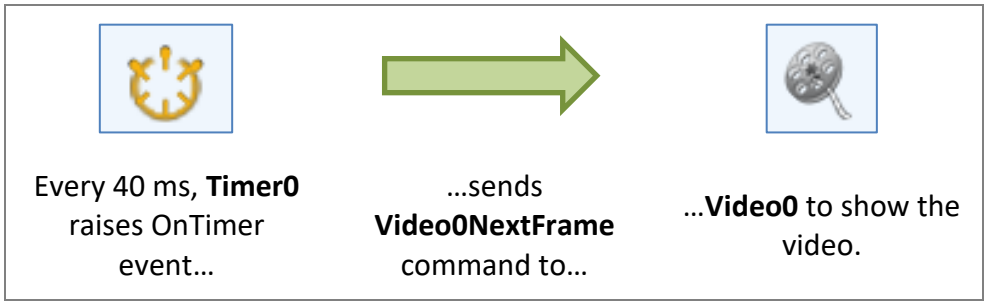
…sends the **Video0NextFrame** to the **Video0** object:



The command **Video0NextFrame** stands for *Tell the **Video0** object to show the next frame of the video.*

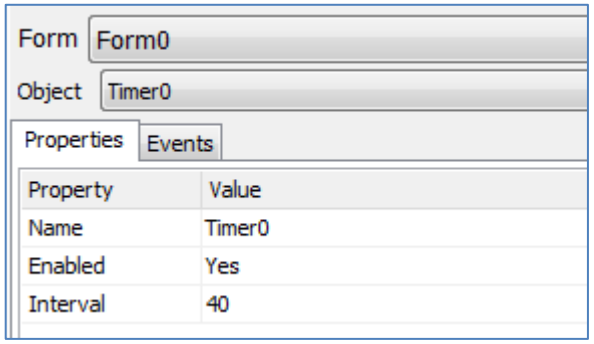
Summarising this indirect Play command gives:



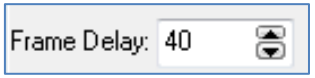


It is important that both the timer and the video have the same value:

- **Interval** for the timer



- And **Frame Delay** for the video:

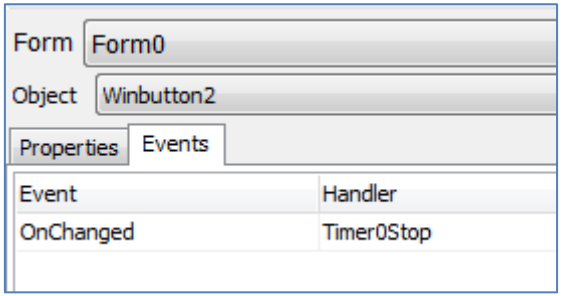


Otherwise, the video may not played at normal speed!

Stop the Video

To stop or pause playing the track, the **WinButton2** button…

…sends the command **Timer0Stop** to **Timer0**:



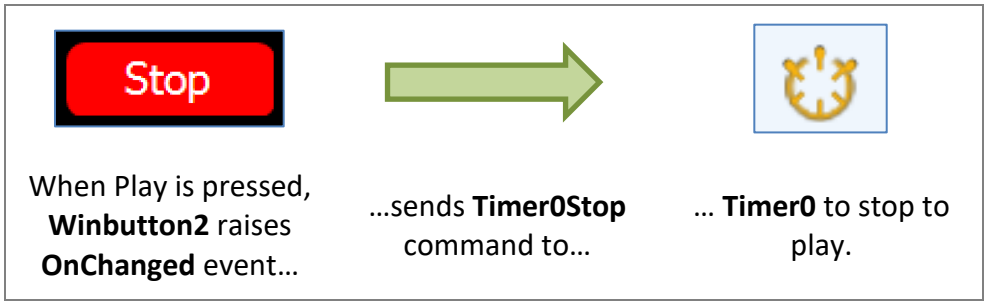
The command **Timer0Stop** stands for *Tell the **Timer0** object to stop the timer and no longer raise events.*

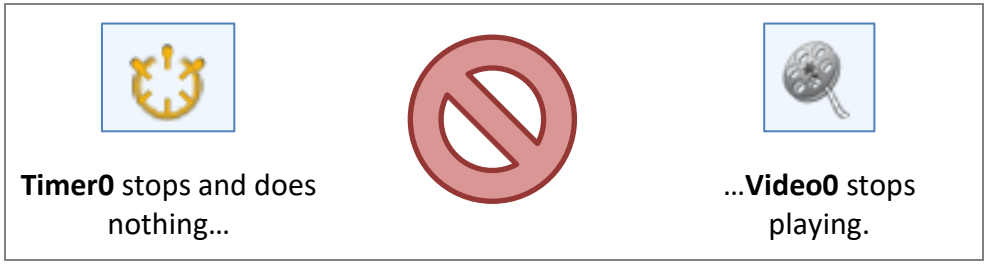
So the **Timer0** timer…

…stops and does nothing.

No more **Video0NextFrame** command is sent to the Video0 object, so the video stops.

Summarising this indirect Stop command gives:





Resume After Stop

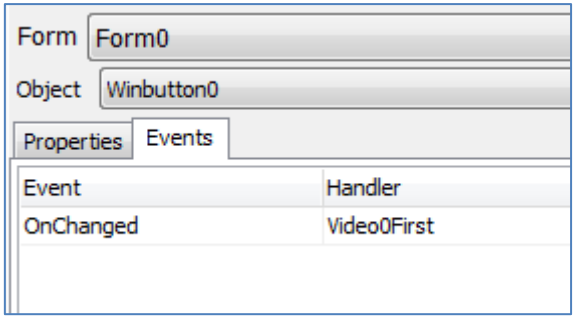
To resume after Stop, just press Play again:

The video resumes playing.

Show First Frame

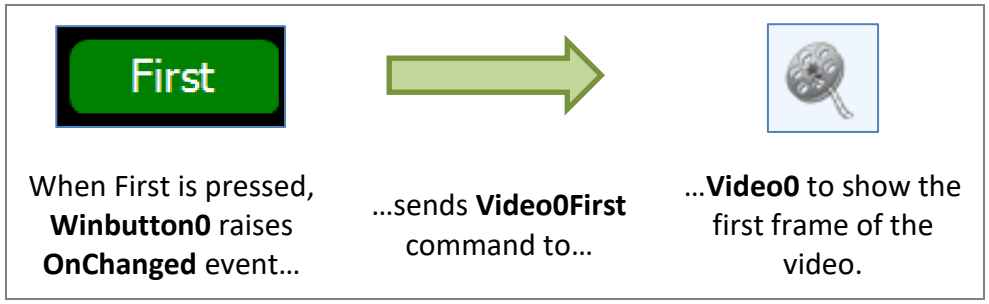
To show the first frame, the **Winbutton0** button…

…sends the command **Video0First** to **Video0**:



The command **Video0Previous** stands for *Tell the **Video0** object to show the previous frame of the video.*

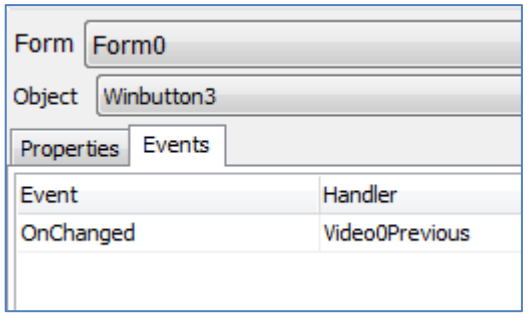
Summarising this direct First command gives:



Show Previous Frame

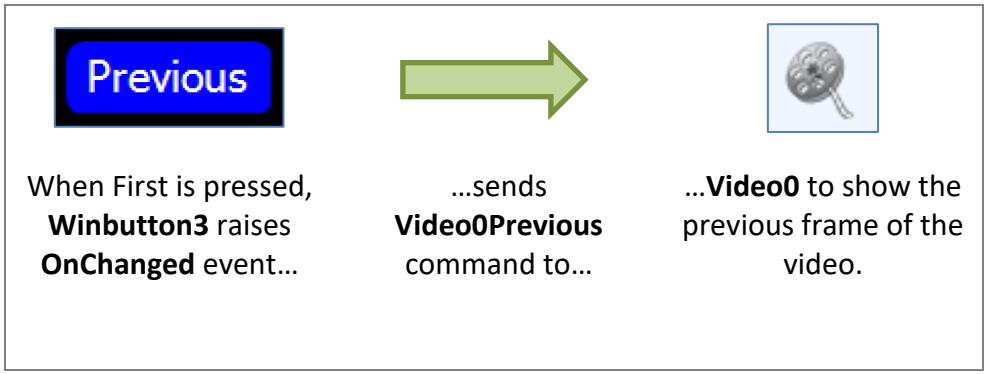
To show the previous frame, the **Winbutton3** button…

…sends the command **Video0Previous** to **Video0**:



The command **Video0Previous** stands for *Tell the **Video0** object to show the previous frame of the video.*

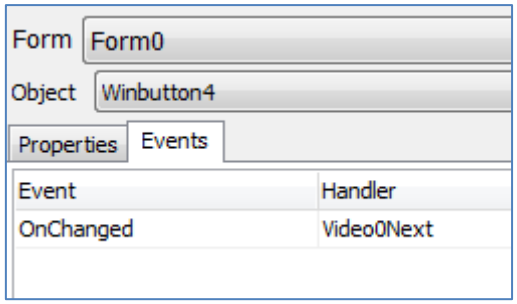
Summarising this direct Previous command gives:



Show Next Frame

To show the next frame, the **Winbutton4** button…

…sends the command **Video0Next** to **Video0**:



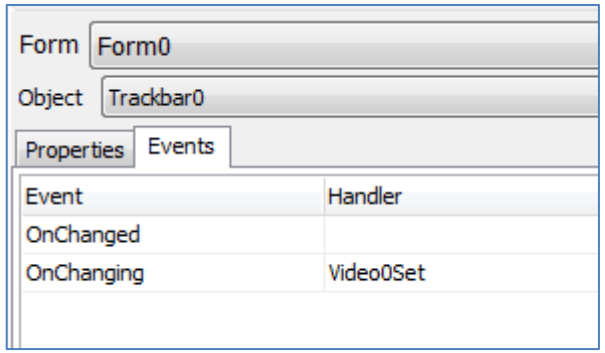
The command **Video0Next** stands for *Tell the **Video0** object to show the next frame of the video.*

Summarising this direct Next command gives:

Select the Frame

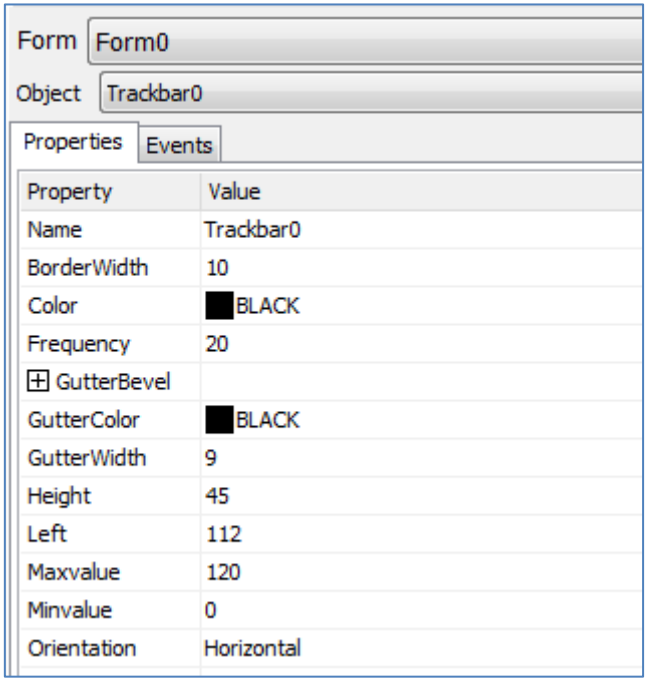
To select a specific frame, just move the slider on track-bar, the **TrackBar0** object…

…raises the event **OnChanging** and sends the command **Video0Set** to **Video0**:



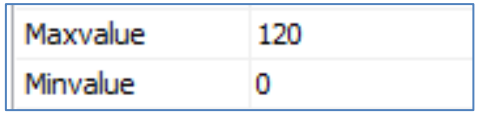
The command **Video0Set** stands for *Tell the* **Video0** *object to display the frame which number specified by the value.*

The TrackBar0 object needs to be set with the correct parameters:

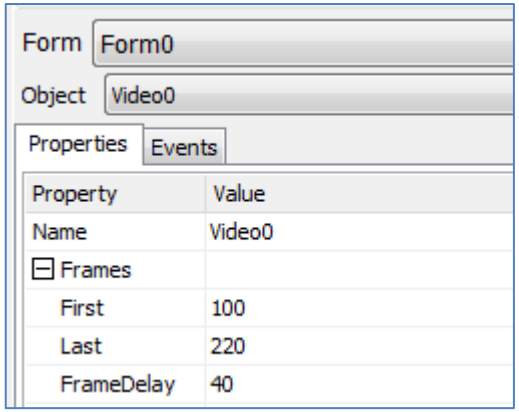


The **MinValue** and **MaxValue** of the **TrackBar** shall be defined as follow:

- **MinValue** is equal to 0 or the first frame to be displayed:



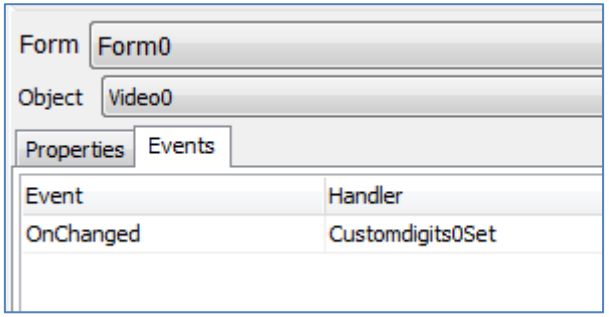
- **MaxValue** corresponds to the duration of the selected sequence, and is equal to the total number of frames selected, here last *220* – first *100* = *120*:



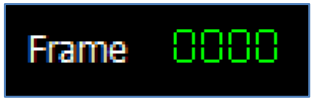
Summarising this command gives:

Display the Frame Number

When **OnChanged** is raised, the **Video0** object sends the command **CustomDigits0Set**…



…and **CustomDigits0** displays the value:



Summarising this command gives:

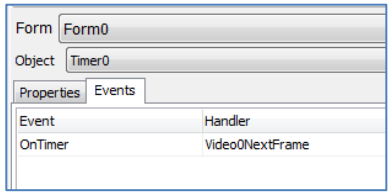
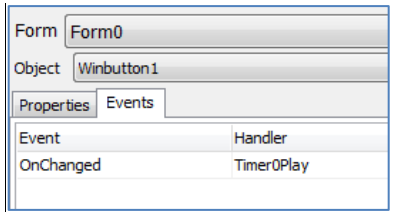
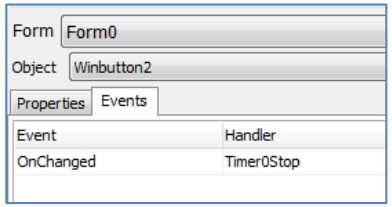
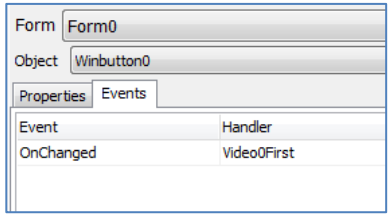
Build and Upload the Project

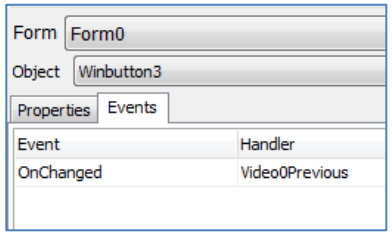
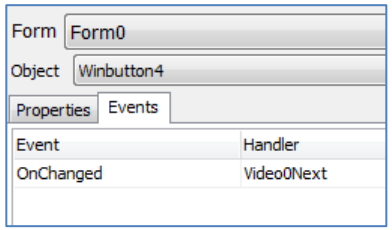
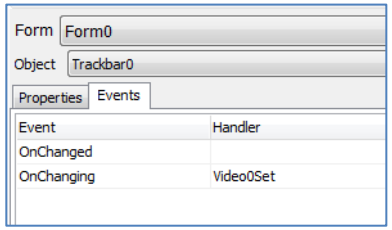
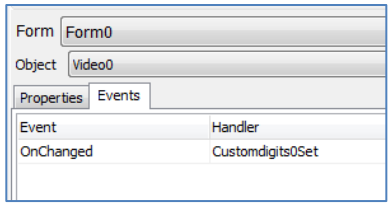
For instructions on how to build and upload a ViSi-Genie project to the target display, please refer to the section "**Build and Upload the Project**" of the application note

ViSi Genie Getting Started – First Project for Picaso Displays
ViSi Genie Getting Started – First Project for Diablo16 Displays

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

Summary

Element	ViSi-Genie	Command	Comment
Timer			**Timer** object sends **Video0NextFrame** command to **Video** object
Play **Resume**			Simple **WinButton** object sends the command **Timer0Play** to **Timer** object. **Timer** object is linked to **Video** object by **Video0NextFrame** command.
Stop			Simple **WinButton** object sends the command **Timer0Stop** to **Timer** object. **Timer** object is linked to **Video** object by **Video0NextFrame** command.
First			Simple **WinButton** object sends the command **Video0First** to **Video** object

Element	ViSi-Genie	Command	Comment
Previous			Simple **WinButton** object sends the command **Video0Previous** to **Video** object
Next			Simple **WinButton** object sends the command **Video0Next** to **Video** object
Select the frame			**TrackBar** object sends the command **Video0Set** to **Video** object
Display frame number			**Video** object sends the command **CustomDigits0Set** to **CustomDigits** object

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.