



Smart Widgets: Slide to Unlock

DOCUMENT DATE: 9th MAY 2020
DOCUMENT REVISION: 1.0



Description

This application note shows how to create a slide-to-unlock widget using a custom horizontal slider on a Picaso, Diablo16 and Pixxi touch screen display modules.

Before getting started, the following are required:

Hardware

- Any [4D Systems display module](#) powered by any of the following processors:
 - o Diablo16
 - o Pixxi28/44
 - o Picaso
- [Programming Adaptor for target display module](#)
- [uSD Card](#)
- USB Card Reader

Software

- [Workshop4](#)
- This requires the **PRO** version of Workshop4.

This application note comes with one (1) ViSi-Genie project.

Note: Using a non-4D programming interface could damage the processor and void the warranty.

Content

Description 2

Content 2

Application Overview..... 3

Setup Procedure 3

Create a New Project 3

Design the Project..... 3

Add a Smart Slider3

Add a Form6

Add a Magic Event8

Link a Smart Slider to a Magic Event.....9

Run the Program.....10

Proprietary Information11

Disclaimer of Warranties & Limitation of Liability11

Application Overview

A smart widget with a slide-to-unlock functionality requires the addition of custom 4DGL code to a normal ViSi-Genie project. In this project, custom 4DGL code was added to the ViSi-Genie project in the form of a Magic Event object. Also, a smart slider object was used as the main widget. Other input objects such as sliders and track bars can also be used.

It is assumed that the reader knows how to create smart widget objects and how to use magic objects. Recommended relevant application notes are:

- [Smart Widgets: Horizontal Slider](#)
- [ViSi-Genie How to Add Magic Objects](#)
- [ViSi-Genie Magic Slide to Unlock](#)

Setup Procedure

For instructions on how to launch Workshop4, 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 Diablo16 Display Modules](#)
- [ViSi-Genie Getting Started - First Project for Picaso Displays](#)
- [ViSi-Genie Getting Started - First Project for Pixxi Display Modules](#)

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

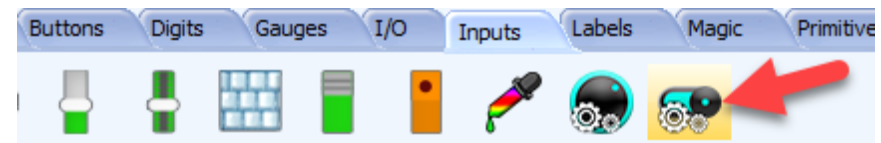
- [ViSi-Genie Getting Started - First Project for Diablo16 Display Modules](#)
- [ViSi-Genie Getting Started - First Project for Picaso Displays](#)
- [ViSi-Genie Getting Started - First Project for Pixxi Display Modules](#)

Design the Project

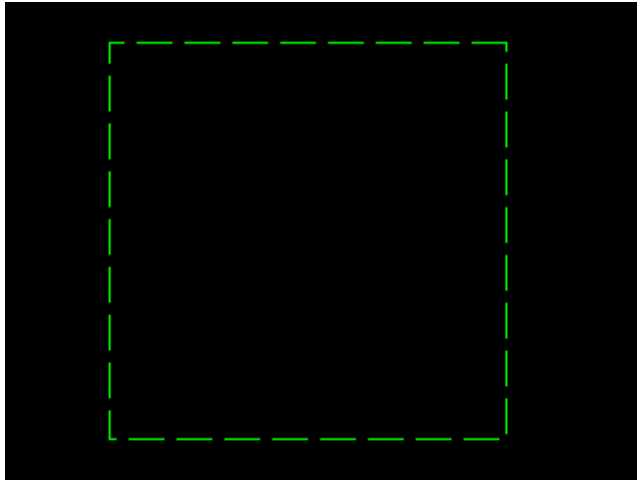
For this application, a gen4-uLCD-32DCT-CLB is used for the project. Same procedure is applicable for any touch screen Picaso, Diablo16 and Pixxi displays.

Add a Smart Slider

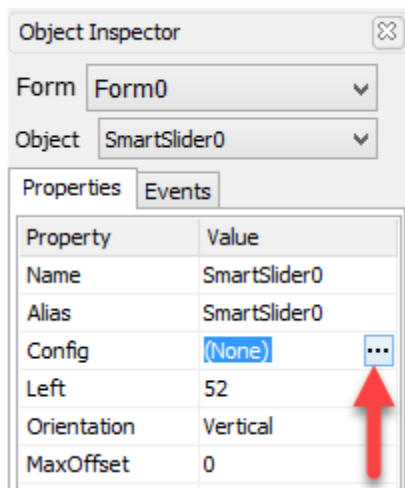
A smart slider object is added to the project by clicking on the smart slider icon under the Inputs menu.



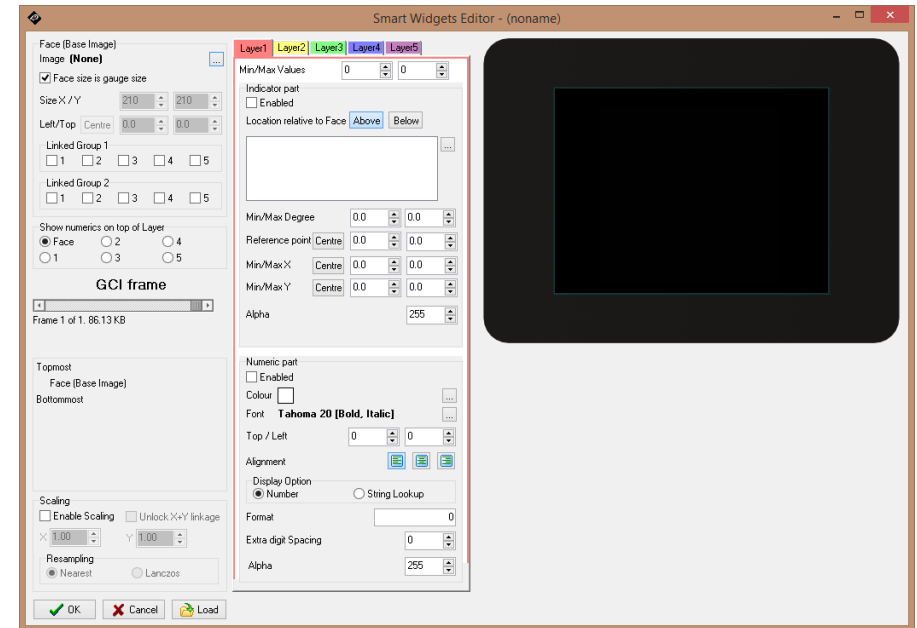
An empty smart slider object is now displayed on the WYSIWYG area. This is **SmartSlider0**.



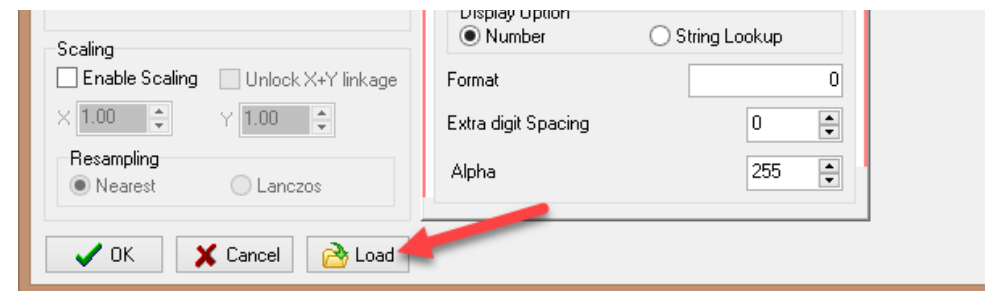
Go to the object inspector and click on the ellipsis dots as shown in the image below.



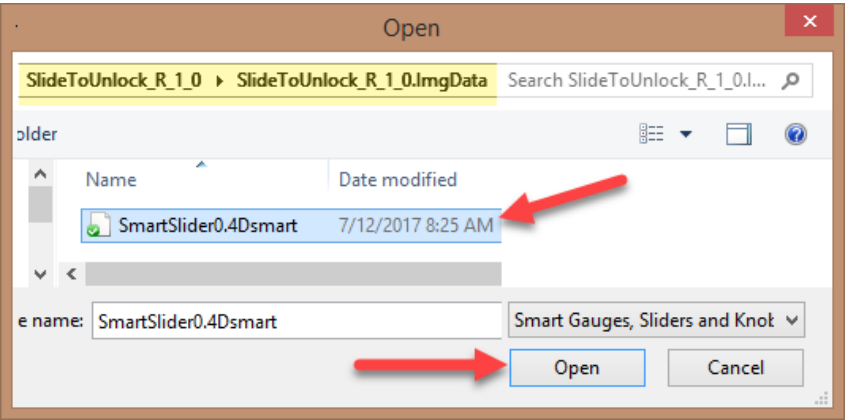
This opens the Smart Widgets Editor tool which allows the user to create or load a smart slider object.



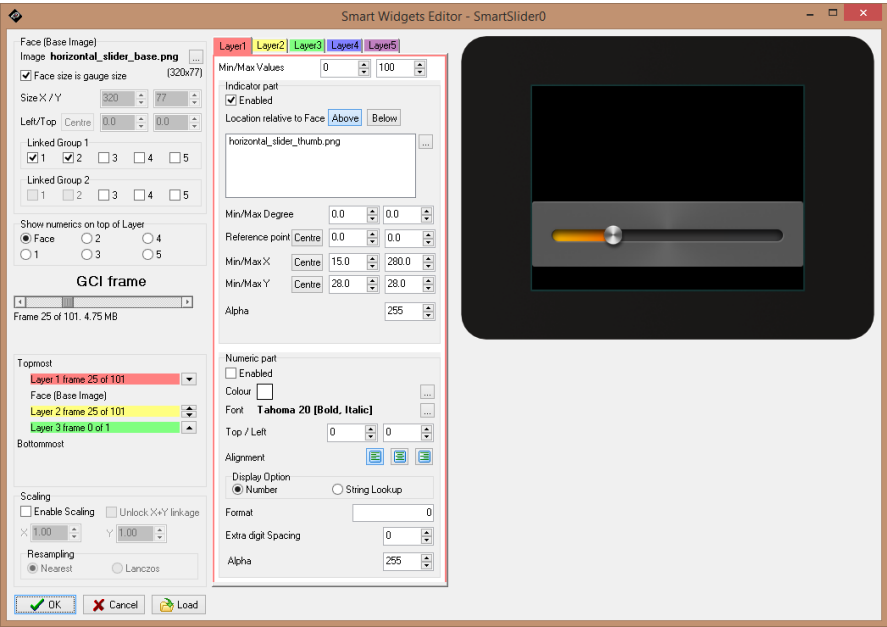
The procedure for creating a smart slider object is detailed in the application note Smart Widgets: Horizontal Slider. An existing smart slider design can be loaded to the project by clicking on the load button.



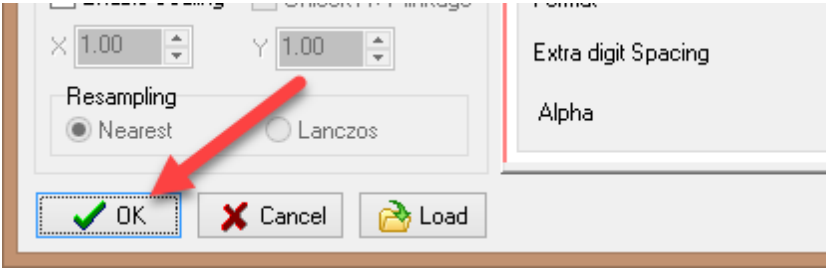
Select the file inside the folder indicated in the image below.



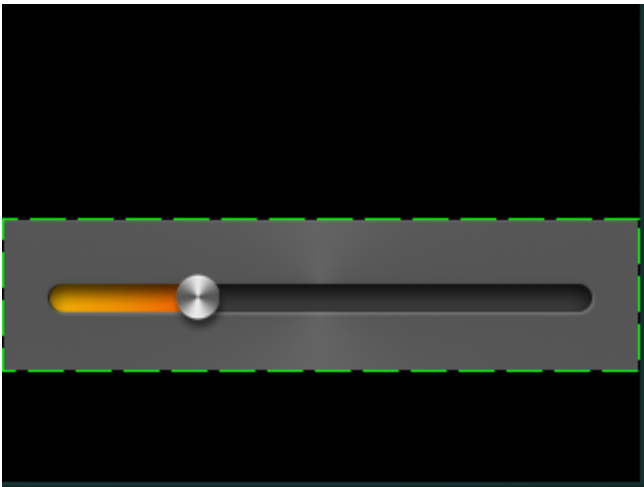
The smart slider object should now appear in the WYSIWYG area of the Smart Widgets Editor tool.



Click OK.



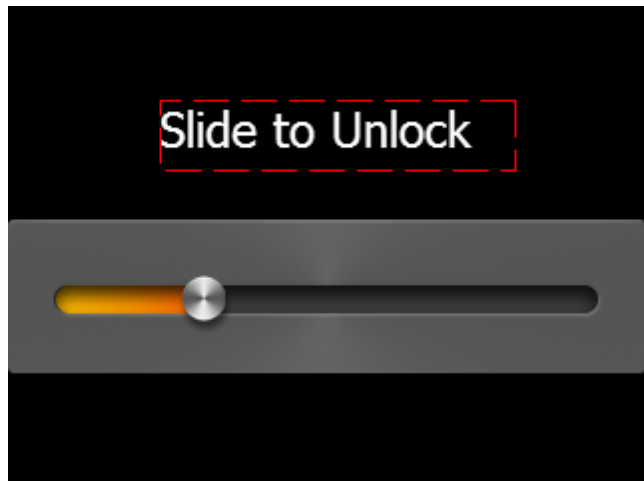
The smart slider object should now appear on the WYSIWYG area. It can be relocated to another position if necessary.



The important properties of **SmartSlider0** are shown below.

Property	Value
Name	SmartSlider0
Alias	SmartSlider0
Config	SmartSlider0
Left	0
Orientation	Horizontal
MaxOffset	30
MinOffset	30
Top	163

A static text object is also added as a label as shown below.

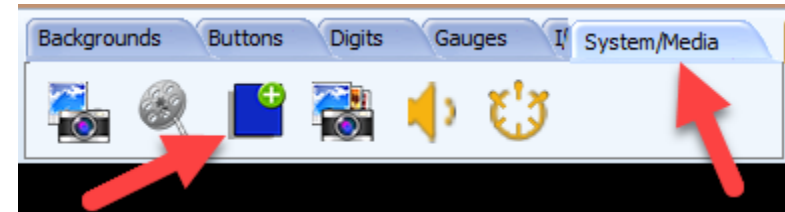


The static text icon is found under the labels menu.



Add a Form

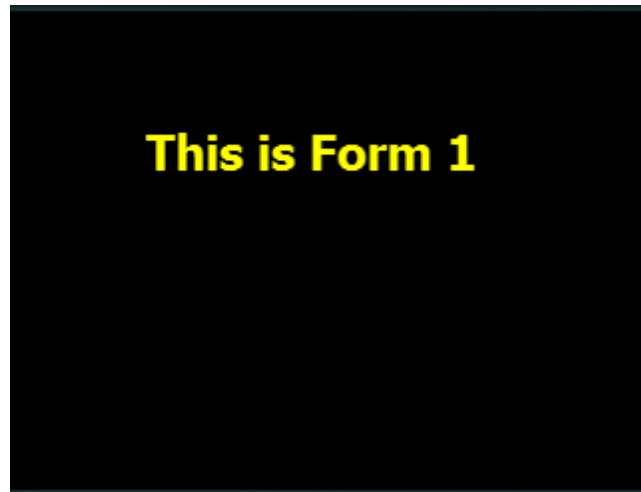
Add another form to the project by clicking on the form icon under the system/media menu.



The new form is **Form1**. The program will navigate to this form when the thumb reaches the right-most part of the slider.



Add a static text object to **Form1** and use it as a label.



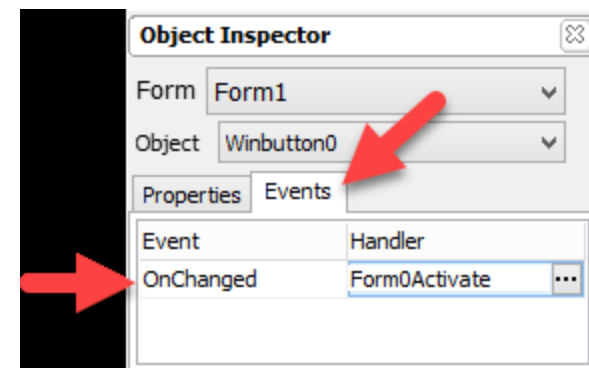
Add and configure a winbutton or fancy button object to be used for navigating back to **Form0**.



This is **Winbutton0**. Change the caption accordingly.

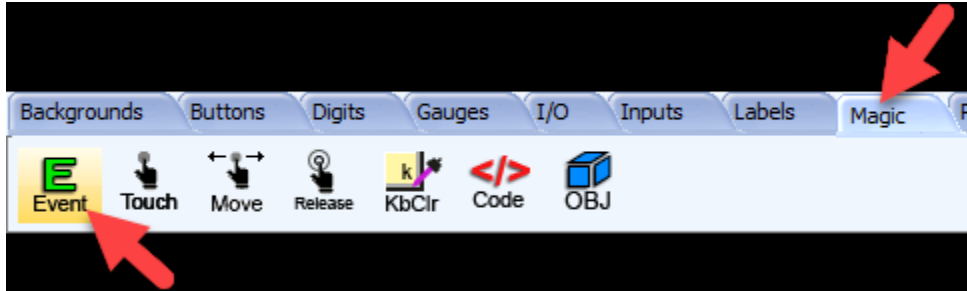


The **OnChanged** event handler of **Winbutton0** should be configured as shown below.

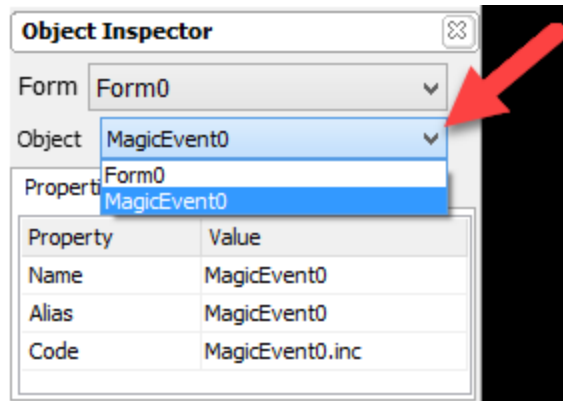


Add a Magic Event

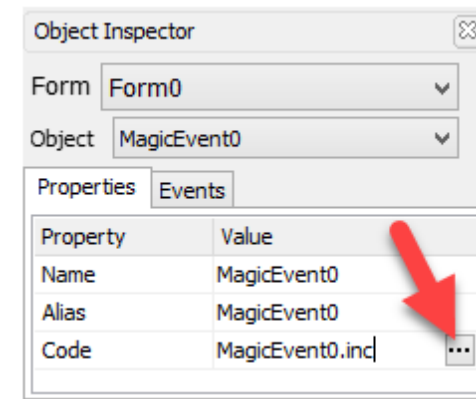
Go to the Magic tab and click on the magic event icon as shown below.



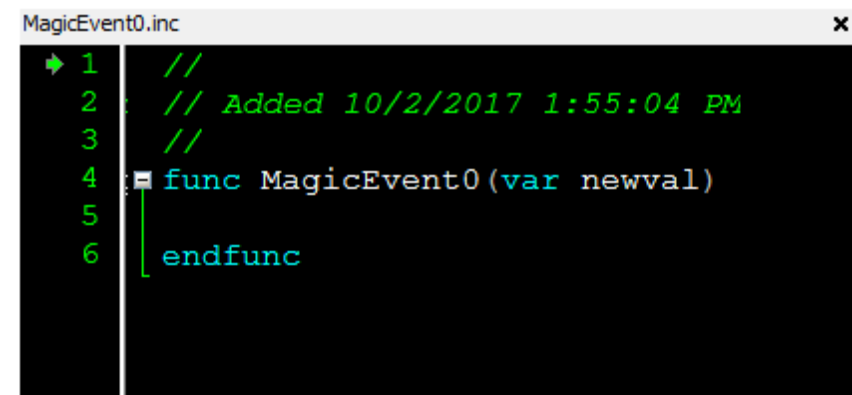
A magic event object called "MagicEvent0" will become available under the dropdown menu of the object inspector as indicated below.



Open the 4DGL code of **MagicEvent0** by clicking on the ellipsis dots as shown below.



The 4DGL code for **MagicEvent0** opens up and shows an empty function.



Open the project attached to this application note and copy the 4DGL code of **MagicEvent0**. The code is shown below.

```

4 func MagicEvent0(var newval)
5   var frame;
6
7   frame := img_GetWord(hndl, iSmartSlider0, IMAGE_INDEX);
8
9   gfx_MoveTo(0,0);
10  print("frame: ", frame, " ");
11
12  if (frame == (img_GetWord(hndl, iSmartSlider0, IMAGE_INDEX) - 1))
13    ActivateForm(1) ;
14  endif
15
16  WriteObject(tSmartSlider, iSmartSlider0, 0) ;
17
18 endfunc

```

It is important to note that **MagicEvent0** is actually a 4DGL function that will be linked to **SmartSlider0**, such that **MagicEvent0** executes every time that **SmartSlider0** is touched and released. The procedure for linking a magic event to a widget is shown in the next section.

The code snippet shown below extracts the current frame of **SmartSlider0**. Note that Genie drives **SmartSlider0** while it is being touched.

```
frame := img_GetWord(hndl, iSmartSlider0, IMAGE_INDEX);
```

The frame number is then printed at the top-left corner of the display module, at x,y coordinate (0, 0).

```
gfx_MoveTo(0,0);
```

```
print("frame: ", frame, " ");
```

An **if** block then checks if the frame number is equal to the highest frame number possible in **SmartSlider0**.

```

if (frame == (img_GetWord(hndl, iSmartSlider0, IMAGE_FRAMES) - 1))
  ActivateForm(1) ;
endif

```

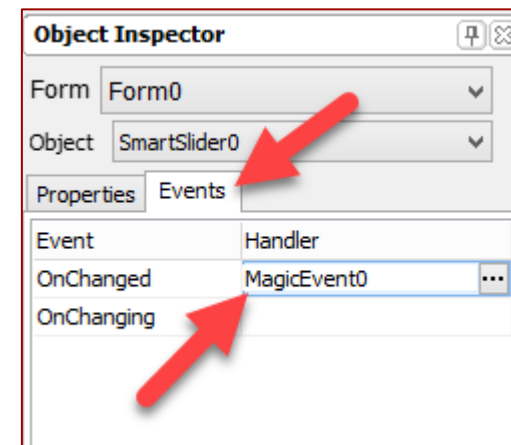
If the above condition is true, **Form1** is activated. **SmartSlider0** is then reset to frame 0, as per the line

```
WriteObject(tSmartSlider, iSmartSlider0, 0) ;
```

This causes the thumb of **SmartSlider0** to slide back to the left-most position.

Link a Smart Slider to a Magic Event

For **SmartSlider0** to be able to trigger **MagicEvent0**, link its **OnChanged** event handler to **MagicEvent0** as shown below.



Run the Program

For instructions on how to save a **ViSi Genie** project, how to connect the target display to the PC, how to select the program destination, and how to compile and download a program, please refer to the section “**Run the Program**” of any of the following application notes:

- [ViSi-Genie Getting Started - First Project for Diablo16 Display Modules](#)
- [ViSi-Genie Getting Started - First Project for Picaso Displays](#)
- [ViSi-Genie Getting Started - First Project for Pixxi Display Modules](#)

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.