CamMonitor.ocx ActiveX Control Guide

1. Introduction 3
2. How to install CamMonitor.ocx 3
3. CamMonitor.ocx parameters 3
4. CamMonitor.ocx methods 7
5. CamMonitor.ocx events 9

Introduction

On the page:

- Purpose of the document
- General information
- Requirements to developers

Purpose of the document

The document contains the information about how to work with CamMonitor.ocx component of ActiveX. The document contains the information about:

1. How to install CamMonitor.ocx.
2. CamMonitor.ocx parameters.
3. CamMonitor.ocx methods.
4. CamMonitor.ocx events.

General information

CamMonitor.ocx is the component of ActiveX that is similar in every way to the **Video monitor interface object.** It allows you to manage cameras, view the archive, etc.

CamMonitor.ocx component supports operation in demo mode.

Requirements to developers

To use CamMonitor.ocx you will need:

1. The knowledge of any programming language that supports using the Component Object Model (COM);
2. Basic knowledge of Win32 programming;
3. Programming environment that supports OCX files.

How to install CamMonitor.ocx

Use the CamMonitorInstaller.exe file (stored in the <*Intellect* installation directory>\Redist folder) to install CamMonitor.ocx.

When installing on 32-bit system, the cammonitor.ocx file is located in the WINDOWS\system32 folder, when installing on 64-bit system – in the WINDOWS\SysWOW64 folder. It is als o registered as the ActiveX component.

CamMonitorInstaller.exe installs the required files for all users.

Besides the library itself, CodecPack driver pack and ITV VideoPlayer tool are installed. ITV VideoPlayer tool uses the CamMonitor.ocx component and allows viewing the archive from the selected camera. By default this tool is installed in the C:\Program Files\ITV VideoPlayer\ folder. The tool interface looks like the one of Converter.exe, but some features of Converter.exe (not dealing with viewing the archive) are not available. This tool can be used to check if CamMonitor.ocx is installed correctly.

CamMonitor.ocx parameters

On the page:

- CamMenuOptions
- CamMenuProcessingOptions
- CamButtonsOptions
- MainPanelOptions
- KeysOptions
- OverlayMode
- How to use parameters

The parameters used for setting the CamMonitor component are presented in this chapter: elements to be shown as well as the overlay mode.

All parameters are *long*-like integers.

The values of parameters used for interface setup are enlisted in the tables and formed in a way that one integer only is represented in binary notation. To set the value of a parameter, combine the values of parameters using the XOR operation. You'll get the number the positions of which in binary notation represent the interface elements that are to be shown and those to be hidden. See How to use parameters section.

The OverlayMode parameter differs from others: it possesses values from 0 to 2 and its value sets the overlay mode.

CamMenuOptions

CamMenuOptions : long

Allows setting the feature menu of the camera.

One or more checkboxes can be set checked.

Available values:

Value	**Information**
#define MENU_ENABLE_OPTION 0x00000001	Show the menu
#define MENU_ARM_ENABLE_OPTION 0x00000002	Show the **Arm** option
#define MENU_REC_ENABLE_OPTION 0x00000004	Show the **Start recording** option
#define MENU_CAMS_ENABLE_OPTION 0x00000008	Show the **Camera** option
#define MENU_TITLES_ENABLE_OPTION 0x00000010	Show the **Show titles** option
#define MENU_PROCESSING_ENABLE_OPTION 0x00000020	Show the **Processing** option
#define MENU_EXPORT_ENABLE_OPTION 0x00000040	Show the **Export** option

CamMenuProcessingOptions

CamMenuProcessingOptions : long

Allows setting the **Processing** menu in the feature menu of the camera.

One or more checkboxes can be set checked:

Available values:

Value	Information
#define MENU_PROCESSING_DEINTERLACE_ENABLE_OPTION 0x00000001	Show the **Deinterlacing** option
#define MENU_PROCESSING_ZOOM_ENABLE_OPTION 0x00000002	Show the **Zoom-in** option
#define MENU_PROCESSING_CONTRAST_ENABLE_OPTION 0x00000004	Show the **Contrast** option
#define MENU_PROCESSING_MASK_ENABLE_OPTION 0x00000008	Show the **Detector mask** option

CamButtonsOptions

CamButtonsOptions : long

Set up displaying the buttons of the CamMonitor component.

One or more checkboxes can be set checked.

Available values:

Value	Information
#define BUTTON_ARCH_ENABLE_OPTION 0x00000001	Show the Archive button
#define BUTTON_TIME_ENABLE_OPTION 0x00000002	Show time
#define BUTTON_NAME_ENABLE_OPTION 0x00000004	Show camera name
#define BUTTON_MENU_ENABLE_OPTION 0x00000008	Show the Menu button
#define BUTTON_RAYS_ENABLE_OPTION 0x00000010	Not used
#define BUTTON_MICS_ENABLE_OPTION 0x00000020	Not used

MainPanelOptions

MainPanelOptions : long

Set up displaying the CamMonitor panel.

One or more checkboxes can be set checked.

Available values:

Value	Information
#define MAIN_PANEL_ENABLE_OPTION 0x00000001	Show the panel

KeysOptions

KeysOptions : long

It allows setting control over the component using the keyboard and mouse.

One or more checkboxes can be set checked.

Available values:

Value	Description
#define TELEMETRY_DISABLE_OPTION 0x00000001	Disables control over the CamMonitor component using the hotkeys available for Video Monitor (see Video Monitor).
#define TELEMETRY_DISABLE_OPTION 0x00000002	Disables Telemetry control using the CamMonitor component (see Telemetry control).

OverlayMode

OverlayMode : long

Set the overlay mode.

Available values:

Value	Information
0	Overlay is not in use
1	Overlay 1
2	Overlay 2

How to use parameters

```
DWORD options = CamMonitor1->CamMenuOptions;
options = options^MENU_CAMS_ENABLE_OPTION^MENU_ARM_ENABLE_OPTION^MENU_REC_ENABLE_OPTION;
CamMonitor1->CamMenuOptions = options;
CamMonitor1->CamMenuProcessingOptions ^= MENU_PROCESSING_MASK_ENABLE_OPTION;
```

CamMonitor.ocx methods

On the page:

- Connect
- ShowCam
- DoReactMonitor
- RemoveAllCams
- IsConnected
- GetCurIp
- SendRawMessage
- Disconnect
- SetCallBackOptions

Connect

Connect(BSTR **ip**, BSTR **login**, BSTR **password**, BSTR **arch_password**, long **param**) set up a connection to the Server.

- BSTR **ip** – IP address of the Video server;
- BSTR **login** – login to connect to the Server (can be blank);
- BSTR **password** – password to set up a connection to the Video server (can be blank);
- BSTR **arch_password** – password to access the archive (i.e. admin password, can be blank);
- long **param** – Server role:
  - 0 – video server;
  - 1 – backup archive;
  - 2 – videogate;

The connection to Server is set up **asynchronously**.

ShowCam

ShowCam(long **cam_id**, long **compress**, long **show**) shows/hides camera on the monitor.

- long **cam_id** – camera ID
- long **compress** – level of video compression 0-5 (for local camera =0)
- long **show** – checkbox: show/hide camera (1/0)

DoReactMonitor

DoReactMonitor(BSTR **react_string**) – control over the monitor/cameras

- BSTR **react_string** – reaction string view

How to create react_string:

```
react_string =  "MONITOR||ARCH_FRAME_TIME|cam<3>,date<dd-mm-yy>,time<hh:mm:ss>";
CamMonitor1->DoReactMonitor(react_string);
```

The result of calling the function with the parameter: camera 3 will be in the archive mode and the archive will be positioned to date «dd-mm-yy» and time «hh:mm:ss» (date and time are to be set in this format only).

The mode parameter takes the following values:

0 – video gate if it's specified (otherwise, video server).

1 – video server.

2 – long-time archive.

"MONITOR|**<id ignored>**|ARCH_FRAME_TIME|..."

Note:
Positioning accuracy can be specified in milliseconds, for instance:

DoReactMonitor("MONITOR||ARCH_FRAME_TIME|cam<3>,date<02-10 05>,time<12:12:22.345>

RemoveAllCams

RemoveAllCams() : long – remove all cameras from the monitor

IsConnected

IsConnected() : boolean – method shows if the Video server is connected or disconnected.

GetCurIp

GetCurIp() : BSTR – returns the IP address of Server specified when calling **Connect**.

SendRawMessage

SendRawMessage(BSTR **msg**) – sends the command to be executed to Video server.

- BSTR **msg** – command string view

How to call a function:

```
m_Cam.SendRawMessage("CAM|1|REC");
```

```
m_Cam.SendRawMessage("CAM|1|REC_STOP");
```

```
m_Cam.SendRawMessage("CAM|1|ARM");
```

```
m_Cam.SendRawMessage("CAM|1|DISARM");
```

Disconnect

Disconnect() – disconnect from the Video server.

SetCallBackOptions

SetCallBackOptions(int **cam_id**, int **options**) – sets parameters of getting video from camera.

- int **cam_id** – camera ID (number).
- int **options** – options. The possible values of the **options parameter are:**
  - WithoutVideoFrame = 0x00 – do not send frames from the video module.
  - WithVideoFrame = 0x01 – send frames from the video module.
  - WithExtendedParams = 0x02 – get frames with extended parameters (time, fps, subtitles).
  - WithInformationLayout = 0x04 – display video in the window with control elements (context menu).
  - WithCompressedData = 0x08 – display video in the native format without decompression (if any).
- WithoutDecode = 0x10 – disable video decoding on the server.
- WithoutSubtitles=0x20 – disable subtitles.

Note. The options parameter is created the same as parameters of the CamMonitor.ocx component – see CamMonitor.ocx parameters.

CamMonitor.ocx events

OnCamListChange (long **cam_id**, long **action**) – occurs when there is connection with the Server or the number of cameras on the Server changes.

• long **cam_id** – camera ID.

• long **action** equals 1, if camera with **id** == **cam_id** exists, otherwise **action** == **0.**

This event occurs as many times as there are cameras on the Server. The negative value of the **cam_id** parameter (cam_id < 0) shows that **OnCamListChange** is not called.

If there are 3 cameras (1, 2, 3) on the Server, then the following events will occur one after another:

CamListChange(1,1)

CamListChange(2,1)

CamListChange(3,1)

CamListChange(-1,1)

Example:

Show the camera with cam_id =2 with **compress =1** compression level;

```
CamMonitor1CamListChange(long cam_id, long action)
{
   if(cam_id == -1)
   {
      CamMonitor1->ShowCam(2,1,1);
   }
}
```