Thermoeye Inc.

THERMOEYE

TMC Series

Camera Control SDK Manual for TMC80 / TMC160 / TMC256

Contact help@thermoeye.co.kr
Technical Support https://github.com/Thermoeye

5F, 169, Sadang-ro, Dongjak-gu, Seoul, Republic of Korea (07003)

Revision

Version	Date	Contents
1.0	NOV.06.2023	Separated from ThermoCam160E/256E User Manual
1.1	NOV.14.2023	Added MatFrame, GetSystemStatus(), GetSystemError()
1.2	NOV.30.2023	Supported input index and coordinates at once for RoiSpot(), RoiLine(), RoiRect(), RoiEllipse()
1.3	JAN.17.2024	Modified product name Added NoiseFiltering
1.4	MAR.07.2024	Added CheckFirmware() Added TMC80 Product Added Turbo, DeepGreen in Color Map
1.5	JUN.03.2024	Added GetNetworkAdapterIPList() Modified GetCameraList() Modified RemoteCamInfo

Chapter

1. ThermoCamSDK C# API

ThemoCamSDK provides the C# APIs to control the functionalities of camera device.

The sample project was created on Microsoft Visual Studio Community 2022 and was implemented on Windows .NET Framework 4.8.

2. `ThermoEngine` Namespace

2.1. Classes

`Camera`	Abstract class for camera control
`RemoteCamera`	Inheritance class for remote camera control
`LocalCamera`	Inheritance class for local camera control
`Frame`	Class for Frame control
`CamInfo`	Abstract class for camera information store
`RemoteCamInfo`	Inheritance class for remote camera information store
`LocalCamInfo`	Inheritance class for local camera information store
`CameraStatus`	Camera device system status class
`RoiManager`	Class for ROI management
`RoiObject`	Abstract class for ROI object
`RoiSpot`	Inheritance class for Spot type ROI
`RoiLine`	Inheritance class for Line type ROI
`RoiRect`	Inheritance class for Rectangle type ROI
`RoiEllipse`	Inheritance class for Ellipse type ROI
`RoiPolygon`	Inheritance class for Polygon type ROI

2.2. Interfaces

`ICameraControl`	Interface for camera sensor control

2.3. Structures

`LocItem`	ROI location and temperature value

2.4. Enumerations

`SysStatusCode`	Camera device system status code
`SysErrorCode`	Camera device system error code
`RoiType`	ROI types
`TempUnit`	Temperature unit type

3. `ThermoEngine.Camera` Class

3.1. Definition

```
public class Camera : IDisposable
```

- The **`Camera`** Class is the main Class in ThermoEngine that provides functionalities to drive and control the Remote (Ethernet Network) and Local (USB) camera. The **`RemoteCamera`** and **`LocalCamera`** Classes are inherited from this **`Camera`** Class and provide functionalities for each connection specification.

- Derived Class

```
public class RemoteCamera : Camera
public class LocalCamera : Camera
```

- Properties

`Name`	Product Name
`Width`	Frame width
`Height`	Frame height
`FPS`	Frame rate, Frames per second
`IsOpen`	State of video streaming, true=play / false=stop
`ColorMap`	Color Map, 0=Grayscale / 1=Autumn / 2=Bone / 3=Jet / 4=Winter / 5=Rainbow / 6=Ocean / 7=Summer / 8=Spring / 9=Cool / 10=Hsv / 11=Pink / 12=Hot / 13=Parula / 14=Magma / 15=Inferno / 16=Plasma / 17=Viridis / 18=Cividis / 19=Twilight / 20=TwilightShifted / 21=Turbo / 22=DeepGreen
`TempUnit`	Temperature unit type, 0=Raw / 1=Celsius / 2=Fahrenheit / 3=Kelvin
`TempUnitSymbol`	Temperature unit symbol
`NoiseFiltering`	Video noise filtering on/off

- Methods

`Open(LocalCamInfo)`	Opens local camera device connection via USB UVC and CDC
`Open(RemoteCamInfo)`	Opens remote camera device connection via Ethernet RTSP and RTP
`Close`	Closes camera device connection
`QueryFrame`	Queries a resized frame
`GetTemperature`	Get converted temperature by temperature unit
`Start`	Starts camera video streaming
`Stop`	Stops camera video streaming

`Dispose`	Cleans up resources being used

- Fields

`Control`	Prepares camera control interface.

3.2. `Open(LocalCamInfo)` Method

```
public bool Open(LocalCamInfo camInfo)
```

- Opens local camera device connection via USB UVC and CDC
- Parameters

  `camInfo`: device information to be connected via USB

- Return Value

  True if this method opens camera connection successfully; otherwise, false if an exception is raised.

3.3. `Open(RemoteCamInfo)` Method

```
public bool Open(RemoteCamInfo camInfo)
```

- Opens remote camera device connection via Ethernet RTSP and RTP
- Parameters

  `camInfo`: device information to be connected via Ethernet

- Return Value

  True if this method opens camera connection successfully; otherwise, false if an exception is raised.

3.4. `Close` Method

```
public bool Close()
```

- Closes camera device connection
- Return Value

  True if this method closes camera connection successfully; otherwise, false if an exception is raised.

3.5. `QueryFrame` Method

```
public Frame QueryFrame(int width, int height)
```

- Queries a resized frame
- Parameters

  `width`: desired width

  `height`: desired height
- Return Value

  Frame object if this method gets a captured frame successfully; otherwise, null if an exception is raised.
- Remarks

  If you want to get the original frame object, just call method without parameters as below:

  ```
  var frame = mCamera.QueryFrame();
  ```

3.6. `GetTemperature` Method

```
public double GetTemperature(double raw)
```

- Get converted temperature by temperature unit
- Parameters

  `raw`: raw value to be converted
- Return Value

  Converted temperature value

3.7. `Start` Method

```
public bool Start()
```

- Starts camera video streaming
- Return Value

  True if this method starts camera video streaming successfully; otherwise, false if an exception is raised.

3.8. `Stop` Method

```
public bool Stop()
```

- Stops camera video streaming
- Return Value

  True if this method stops camera video streaming successfully; otherwise, false if an exception is raised.

3.9. `Dispose` Method

```
public void Dispose()
```

- Cleans up resources being used

3.10. `Control` Field

```
public ICameraControl Control
```

- Prepares camera control interface
- Must use this field to call camera control methods as below:

```
mCamera.Control.GetSensorModelName();
```

4. `ThermoEngine.RemoteCamera` Class

4.1. Definition

```
public class RemoteCamera : Camera
```

- The **`RemoteCamera`** Class is inherited from the **`Camera`** Class and provides functionalities to drive and control a Remote (Ethernet Network) camera.
- Constructors

`RemoteCamera`	Initializes a new instance of the **`RemoteCamera`** class

- Additional Methods

`GetCameraList`	Gets remote camera list
`GetNetworkAdapterIPList`	Gets local network adapter IP address list

4.2. `RemoteCamera` Constructor

```
public RemoteCamera()
```

- Initializes a new instance of the `RemoteCamera` class

4.3. `GetCameraList` Method

```
static public RemoteCamInfo[] GetCameraList(string adapterIP = null)
```

- Gets remote camera list
- Parameters

  `adapterIP`: specified local network adapter IP address

- Return Value

  Camera information list if this method gets connectable camera information successfully; otherwise, false if an exception is raised.

- Remark

  If you want to get all cameras that can be connected via multiple local network adapters, just call method without parameters as below:

  ```
  object list = RemoteCamera.GetCameraList();
  ```

4.4. `GetNetworkAdapterIPList` Method

```
static public string[] GetNetworkAdapterIPList()
```

- Gets local network adapter IP address list
- Return Value

  IP address list if this method gets active local network adapters successfully; otherwise, false if an exception is raised.

5. `ThermoEngine.LocalCamera` Class

5.1. Definition

```
public class LocalCamera : Camera
```

- The `LocalCamera` Class is inherited from the `Camera` Class and provides functionalities to drive and control a Local (USB) camera.

- Constructors

LocalCamera	Initializes a new instance of the `LocalCamera` class

- Additional Methods

GetCameraList	Gets local camera list

5.2. `LocalCamera` Constructor

```
public LocalCamera()
```

- Initializes a new instance of the `LocalCamera` class

5.3. `GetCameraList` Method

```
static public LocalCamInfo[] GetCameraList()
```

- Gets local camera list
- Return Value

  Camera information list if this method gets connectable camera information successfully; otherwise, false if an exception is raised.

6. `ThermoEngine.Frame` Class

6.1. Definition

```
public class Frame : IDisposable
```

- The `Frame` Class provides functionalities to convert a captured single frame data object to a bitmap image object.
- Methods

GetPixel(int, int)	Gets pixel data in frame
GetPixel(int, int, int, int)	Gets area pixel data in frame
SetPixel(int, int, ushort)	Sets raw data into frame
SetPixel(int, int, int, int, ushort)	Sets raw data into area in frame
MinMaxLoc	Gets minimum, maximum and average values and locations
DoMeasure(RoiObject)	Measures location and temperature by ROI
DoMeasure(List<RoiObject>)	Measures location and temperature by ROI list
ToBitmap	Converts a captured frame to Bitmap image object

Dispose	Releases all resources used by the Frame

- Fields

MatFrame	Mat array for a captured frame

6.2. GetPixel(int, int) Method

```
public double GetPixel(int x, int y)
```

- Gets pixel data in frame
- Parameters

  x: x position

  y: y position
- Return Value

  pixel data value

6.3. GetPixel(int, int, int, int) Method

```
public double[,] GetPixel(int x, int y, int width, int height)
```

- Gets area pixel data in frame
- Parameters

  x: x position

  y: y position

  width: width

  height: height
- Return Value

  pixel data values

6.4. SetPixel(int, int, ushort) Method

```
public bool SetPixel(int x, int y, ushort value)
```

- Sets pixel data into frame
- Parameters

`x`: x position

`y`: y position

`value`: raw value

- Return Value

  True if this method set value successfully; otherwise, false if an exception is raised.

6.5. `SetPixel(int, int, int, int, ushort)` Method

```
public bool SetPixel(int x, int y, int width, int height, ushort value)
```

- Sets pixel data into area in frame
- Parameters

  `x`: x position

  `y`: y position

  `width`: width

  `height`: height

  `value`: raw value

- Return Value

  True if this method set value successfully; otherwise, false if an exception is raised.

6.6. `MinMaxLoc` Method

```
public void MinMaxLoc(out double minVal,
                      out double avgVal,
                      out double maxVal,
                      out System.Drawing.Point minLoc,
                      out System.Drawing.Point maxLoc)
```

- Gets minimum, maximum and average values and locations
- Parameters

  `minVal`: minimum value

  `avgVal`: average value

  `maxVal`: maximum value

  `minLoc`: minimum location

`maxLoc`: maximum location

6.7. `DoMeasure(RoiObject)` Method

```
public void DoMeasure(ref RoiObject item)
```

- Measures location and temperature by ROI
- Parameters

  `item`: measured ROI object

6.8. `DoMeasure(List<RoiObject>)` Method

```
public void DoMeasure(ref List<RoiObject> items)
```

- Measures location and temperature by ROI list
- Parameters

  `item`: measured ROI objects list

6.9. `ToBitmap` Method

```
public Bitmap ToBitmap(int width, int height)
```

- Converts a captured frame to Bitmap image object
- Parameters

  `width`: width size to be resized

  `height`: height size to be resized

- Return Value

  Bitmap image object if this method converts a frame successfully; otherwise, null if an exception is raised.

6.10. `Dispose` Method

```
public void Dispose()
```

- Releases all resources used by the Frame

7. ThermoEngine.CamInfo Class

7.1. Definition

```
public class CamInfo
```

- The **CamInfo** Class is an abstraction Class to store information about connected Remote (Ethernet Network) or Local (USB) camera. The **RemoteCamInfo** and **LocalCamInfo** Classes are inherited from this **CamInfo** Class to store their connection information.
- Derived Class :

```
public class RemoteCamInfo : CamInfo
public class LocalCamInfo : CamInfo
```

8. ThermoEngine.RemoteCamInfo Class

8.1. Definition

```
public class RemoteCamInfo : CamInfo
```

- The **RemoteCamInfo** Class is inherited from the **CamInfo** Class and provides functionalities to store information about a connected Remote (Ethernet Network) camera.
- Constructors

RemoteCamInfo	Initializes a new instance of the **RemoteCamInfo** class

- Fields

Name	Camera device name
SerialNumber	Product serial number
AddrMAC	MAC address
AddrIP	IP address
AdapterIP	Local Network Adapter IP address

9. ThermoEngine.LocalCamInfo Class

9.1. Definition

```
public class LocalCamInfo : CamInfo
```

- The **LocalCamInfo** Class is inherited from the **CamInfo** Class and provides functionalities to store information about a connected Local (USB) camera.
- Constructors

`LocalCamInfo`	Initializes a new instance of the `LocalCamInfo` class

- Fields

`Index`	Camera index
`Name`	Camera device name
`ComPort`	Serial port name

10. `ThermoEngine.RoiManager` Class

10.1. Definition

```
public class RoiManager
```

- The `RoiManager` Class provides functionalities to manage objects by ROI type.
- Constructors

`RoiManager`	Initializes a new instance of the `RoiManager` class

- Properties

`SelectedItem`	Selected ROI object
`SelectedType`	Selected ROI type

- Methods

`Clear`	Clear ROI object
`MouseDown`	Mouse down event handler
`MouseMove`	Mouse move event handler
`MouseUp`	Mouse up event handler

- Fields

`Items`	ROI object list
`roiCount`	ROI creation count up

10.2. `Clear` Method

```
public void Clear()
```

- Clear ROI object

10.3.`MouseDown` Method

```
public bool MouseDown(object sender, Point pt)
```

- Mouse down event handler
- Parameters

  `sender`: event sender

  `pt`: point coordinates
- Return Value

  True if this method does successfully; otherwise, false if an exception is raised.

10.4.`MouseMove` Method

```
public bool MouseMove(object sender, Point pt)
```

- Mouse move event handler
- Parameters

  `sender`: event sender

  `pt`: point coordinates
- Return Value

  True if this method does successfully; otherwise, false if an exception is raised.

10.5.`MouseUp` Method

```
public bool MouseUp(object sender, Point pt)
```

- Mouse up event handler
- Parameters

  `sender`: event sender

  `pt`: point coordinates
- Return Value

  True if this method does successfully; otherwise, false if an exception is raised.

11. `ThermoEngine.RoiObject` Class

11.1. Definition

```
public class RoiObject
```

- The `RoiObject` Class provides functionalities to control objects by ROI type. The `RoiSpot`, `RoiLine`, `RoiRect`, `RoiEllipse`, `RoiPolygon` Classes are inherited from this `RoiObject` Class and provides functionalities for each type.
- Constructors

`RoiObject`	Initializes a new instance of the `RoiObject` class

- Fields

`Index`	ROI object index
`RoiType`	ROI type, Hand=0 / Spot=1 / Line=2 / Rect=3 / Ellipse=4 / Polygon=5
`MinLoc`	Location for minimum temperature in ROI
`AvgLoc`	Location for average temperature in ROI
`MaxLoc`	Location for maximum temperature in ROI

12. `ThermoEngine.RoiSpot` Class

12.1. Definition

```
public class RoiSpot
```

- The `RoiSpot` Class is inherited from the `RoiObject` Class and provides functionalities to control Spot type ROI object.
- Constructors

`RoiSpot()`	Constructor of RoiSpot
`RoiSpot(int)`	Constructor of RoiSpot by object index
`RoiSpot(Point)`	Constructor of RoiSpot by point coordinates
`RoiSpot(int, int)`	Constructor of RoiSpot by x & y-coordinates
`RoiSpot(int, int, int)`	Constructor of RoiSpot by object index and x & y-coordinates

- Fields

`Spot`	Coordinates of Spot

13. ThermoEngine.RoiLine Class

13.1. Definition

```
public class RoiLine
```

- The `RoiLine` Class is inherited from the `RoiObject` Class and provides functionalities to control Line type ROI object.
- Constructors

`RoiLine()`	Constructor of RoiLine
`RoiLine(int)`	Constructor of RoiLine by object index
`RoiLine(Point, Point)`	Constructor of RoiLine by start & end point coordinates
`RoiLine(int, int, int, int)`	Constructor of RoiLine by start x & y-coordinates and end x & y-coordinates
`RoiLine(int, int, int, int, int)`	Constructor of RoiLine by object index and start x & y-coordinates and end x & y-coordinates

- Fields

`Start`	Start Coordinates of Line
`Line`	End Coordinates of Line

14. ThermoEngine.RoiRect Class

14.1. Definition

```
public class RoiRect
```

- The `RoiRect` Class is inherited from the `RoiObject` Class and provides functionalities to control Rectangular type ROI object.
- Constructors

`RoiRect()`	Constructor of RoiRect
`RoiRect(int)`	Constructor of RoiRect by object index
`RoiRect(Rectangle)`	Constructor of RoiRect by rectangle location and size
`RoiRect(int, int, int, int)`	Constructor of RoiRect by start x & y-coordinates and width & height
`RoiRect(int, int, int, int, int)`	Constructor of RoiRect by object index and start x & y-coordinates and width & height

- Fields

`Rect`	Location and size of Rectangle

15. `ThermoEngine.RoiEllipse` Class

15.1. Definition

```
public class RoiEllipse
```

- The `RoiEllipse` Class is inherited from the `RoiObject` Class and provides functionalities to control Ellipse type ROI object.

- Constructors

`RoiEllipse()`	Constructor of RoiEllipse
`RoiEllipse(int)`	Constructor of RoiEllipse by object index
`RoiEllipse(Rectangle)`	Constructor of RoiEllipse by ellipse location and size
`RoiEllipse(int, int, int, int)`	Constructor of RoiEllipse by start x & y-coordinates and width & height
`RoiEllipse(int, int, int, int, int)`	Constructor of RoiEllipse by object index and start x & y-coordinates and width & height

- Fields

`Ellipse`	Location and size of Ellipse

16. `ThermoEngine.RoiPolygon` Class

16.1. Definition

```
public class RoiPolygon
```

- The `RoiPolygon` Class is inherited from the `RoiObject` Class and provides functionalities to control Polygon type ROI object.

- Constructors

`RoiPolygon()`	Constructor of RoiPolygon
`RoiPolygon(int)`	Constructor of RoiPolygon by object index

- Properties

`Item`	Coordinates list

- Methods

`Add`	Add a point of polygon
`Insert`	Insert a point of polygon
`Remove`	Remove a point of polygon
`RemoveAt`	Remove a point by index of polygon

- Fields

`Points`	Coordinates list of Polygon

17. `ThermoEngine.ICameraControl` Interface

17.1. Definition

```
public interface ICameraControl
```

- The `ICameraControl` Interface provides functionalities to control the sensor of camera device.
- Each method requires the creation of an `ICameraControl` Interface instance and must call the `Control` Field as shown in the example below.

```
mCamera.Control.GetSensorModelName();
```

- Methods

`GetProductModelName`	Gets product model name of camera device
`GetProductSerialNumber`	Gets product serial number of camera device
`GetHardwareVersion`	Gets hardware version of camera device
`GetBootloaderVersion`	Gets bootloader version of camera device software
`GetFirmwareVersion`	Gets firmware version of camera device software
`GetSystemStatus`	Gets system status of camera device
`GetSystemError`	Gets system error of camera device
`GetSensorModelName`	Gets sensor model name of camera sensor
`GetSensorSerialNumber`	Gets sensor serial number of camera device
`GetSensorUptime`	Gets current uptime in milliseconds of camera sensor
`ConvertRawToCelsius`	Converts pixel raw value to Celsius value
`ConvertRawToFahrenheit`	Converts pixel raw value to Fahrenheit value
`ConvertRawToKelvin`	Converts pixel raw value to Kelvin value
`GetNetworkConfiguration`	Gets network configuration of camera device
`SetNetworkConfiguration`	Sets network configuration of camera device
`SetDefaultNetworkConfiguration`	Sets network configuration of camera device to

	factory default values
RebootDevice	Reboot camera device
CheckFirmware	Checks verification of firmware file and read firmware binary information
OpenFirmware	Opens firmware file to update new firmware of camera device
UpdateFirmware	Updates chunk data of firmware binary to camera device
CloseFirmware	Closes opened firmware file

17.2.`GetProductModelName` Method

```
public string GetProductModelName()
```

- Gets product model name of camera device
- Return Value

  Text string if this method gets value from device successfully; otherwise, null if an exception is raised.

17.3.`GetProductSerialNumber` Method

```
public string GetProductSerialNumber()
```

- Gets product serial number of camera device
- Return Value

  Text string if this method gets value from device successfully; otherwise, null if an exception is raised.

17.4.`GetHardwareVersion` Method

```
public string GetHardwareVersion()
```

- Gets hardware version of camera device
- Return Value

  Text string if this method gets value from device successfully; otherwise, null if an exception is raised.

17.5.`GetBootloaderVersion` Method

```
public string GetBootloaderVersion()
```

- Gets bootloader version of camera device software
- Return Value

  Text string if this method gets value from device successfully; otherwise, null if an exception is raised.

17.6.`GetFirmwareVersion` Method

```
public string GetFirmwareVersion()
```

- Gets firmware version of camera device software
- Return Value

  Text string if this method gets value from device successfully; otherwise, null if an exception is raised.

17.7.`GetSystemStatus` Method

```
public Tuple<ushort, string> GetSystemStatus()
```

- Gets system status of camera device
- Return Value

  Status code with message of SysStatus if this method gets status from device successfully; otherwise, 0xFFFF if an exception is raised.

17.8.`GetSystemError` Method

```
public Tuple<ushort, string> GetSystemError()
```

- Gets system error of camera device
- Return Value

  Error code with message of SysError if this method gets status from device successfully; otherwise, 0xFFFF if an exception is raised.

17.9. `GetSensorModelName` Method

```
public string GetSensorModelName()
```

- Gets sensor model name of camera sensor
- Return Value

  Text string if this method gets value from device successfully; otherwise, null if an exception is raised.

17.10. `GetSensorSerialNumber` Method

```
public string GetSensorSerialNumber()
```

- Gets sensor serial number of camera device
- Return Value

  Text string if this method gets value from device successfully; otherwise, null if an exception is raised.

17.11. `GetSensorUptime` Method

```
public string GetSensorUptime()
```

- Gets current uptime in milliseconds of camera sensor
- Return Value

  Text string if this method gets value from device successfully; otherwise, null if an exception is raised.

17.12. `ConvertRawToCelsius` Method

```
public double ConvertRawToCelsius(double rawValue)
```

- Converts raw value to Celsius value
- Parameters

  `rawValue` raw value to be converted
- Return Value

  Celsius value

17.13. `ConvertRawToFahrenheit` Method

```
public double ConvertRawToFahrenheit(double rawValue)
```

- Converts raw value to Fahrenheit value
- Parameters

  `rawValue` raw value to be converted
- Return Value

  Fahrenheit value

17.14. `ConvertRawToKelvin` Method

```
public double ConvertRawToKelvin(double rawValue)
```

- Converts raw value to Kelvin value
- Parameters

  `rawValue` raw value to be converted
- Return Value

  Kelvin value

17.15. `GetNetworkConfiguration` Method

```
public bool GetNetworkConfiguration(
    out string mac, out string ipAssign, out string ip,
    out string netmask, out string gateway, out string dns,
    out string dns2
)
```

- Gets network configuration of camera device
- Parameters

  `mac`: obtained value for MAC address

  `ipAssign`: obtained value for IP assignment, Static or DHCP

  `ip`: obtained value for IP address, IPv4 only

  `netmask`: obtained value for netmask address, IPv4 only

  `gateway`: obtained value for gateway address, IPv4 only

  `dns`: obtained value for main DNS address, IPv4 only

`dns2`: obtained value for sub DNS address, IPv4 only

- Return Value

  True if this method gets values from device successfully; otherwise, false if an exception is raised.

17.16. `SetNetworkConfiguration` Method

```
public bool SetNetworkConfiguration(
    string ipAssign, string ip, string netmask,
    string gateway, string dns, string dns2
)
```

- Sets network configuration of camera device
- Parameters

  `ipAssign`: value to be set for IP assignment, Static or DHCP

  `ip`: value to be set for IP address, IPv4 only

  `netmask`: value to be set for netmask address, IPv4 only

  `gateway`: value to be set for gateway address, IPv4 only

  `dns`: value to be set for main DNS address, IPv4 only

  `dns2`: value to be set for sub DNS address, IPv4 only

- Return Value

  True if this method sets values from device successfully; otherwise, false if an exception is raised.

17.17. `SetDefaultNetworkConfiguration` Method

```
public bool SetDefaultNetworkConfiguration(
    out string ipAssign, out string ip, out string netmask,
    out string gateway, out string dns, out string dns2
)
```

- Sets network configuration of camera device to factory default values
- Parameters

  `ipAssign`: obtained default value for IP assignment, Static or DHCP

  `ip`: obtained default value for IP address, IPv4 only

  `netmask`: obtained default value for netmask address, IPv4 only

`gateway`: obtained default value for gateway address, IPv4 only

`dns`: obtained default value for main DNS address, IPv4 only

`dns2`: obtained default value for sub DNS address, IPv4 only

- Return Value

  True if this method sets values from device successfully; otherwise, false if an exception is raised.

17.18. `RebootDevice` Method

```
public bool RebootDevice()
```

- Reboots camera device
- Return Value

  True if camera device starts reboot successfully; otherwise, false if an exception is raised.

17.19. `CheckFirmware` Method

```
public bool CheckFirmware(
    string fwFilePath, out string vendorName, out string productName,
    out string versionName, out string buildTime, out int fileSize
)
```

- Checks verification of firmware file and read firmware binary information
- Parameters

  `fwFilePath`: firmware file path to be loaded

  `vendorName`: obtained vendor name

  `productName`: obtained product name

  `versionName`: obtained version name

  `buildTime`: obtained build time

  `fileSize`: obtained firmware binary size

- Return Value

  True if the firmware file is suitable for update properly; otherwise, -1 if an exception is raised.

17.20. `OpenFirmware` Method

```
public int OpenFirmware(string fwFilePath)
```

- Opens firmware file to update new firmware of camera device
- Parameters

  `fwFilePath`: firmware file path to be updated

- Return Value

  Binary size if device opens a firmware file successfully; otherwise, -1 if an exception is raised.

17.21. `UpdateFirmware` Method

```
public int UpdateFirmware()
```

- Updates chunk data of firmware binary to camera device
- Return Value

  Percentage value in progress if this method updates chunk data to device successfully; otherwise, -1 if an exception is raised.

17.22. `CloseFirmware` Method

```
public bool CloseFirmware()
```

- Closes opened firmware file
- Return Value

  True if this method closes firmware file successfully; otherwise, false if an exception is raised.

- Remarks

  Device will reboot automatically.