Testo AG

IrApi Documentation

Version 1.6.2
31.07.2013

Testo IrApi Documentation

Table of Content

Testo IrApi Documentation

1 Introduction

1.1 IrApi

The Testo *IrApi* is a program library which makes it possible to access the content of the Testo AG picture format (.bmt) from your own application program.

Main Featurs of *IrApi*:

- Get information like device name, serial number or temperature of a point
- View thermal images
- View visual images

There is **no** possibility to communicate directly with the Testo Cameras or work with video.

1.2 Supported Operation Systems

The *IrApi* supports following operation systems:

- Windows XP SP 3 or higher
- Windows Vista SP 2 or higher
- Windows 7 SP 1 or higher
- Windows 8 or higher

1.3 Supported Languages

IrApi supports following programming languages:

- C
- C#
- VB.Net

2 Installation

The installation consists of two subparts, which automatically starts by clicking *setup.exe*: the installation of prerequisites and the *IrApi* itself.

2.1 Prerequisites

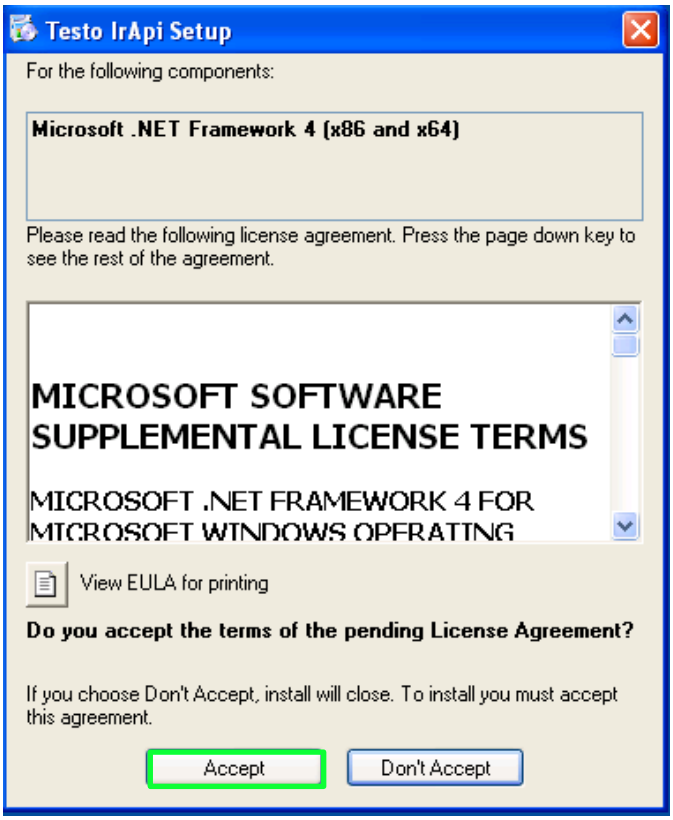
If not already installed, the *IrApi* installer will install following prerequisites:

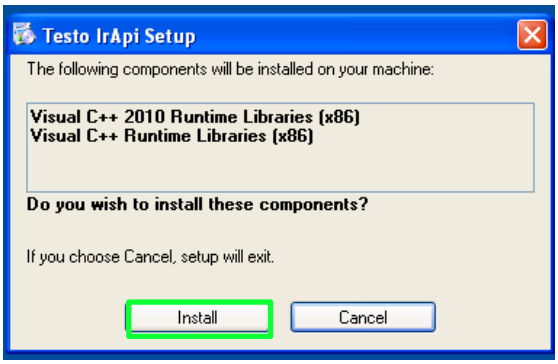
- VCRedist.exe 2008
- VCRedist.exe 2010 x86
- .Net-Framework 4.0

By operation systems based on x64 would additionally add:

- VCRedist.exe 2010 Sp1 x64

You must allow the install of the prerequisites. The Setup will prompt you following dialogs:



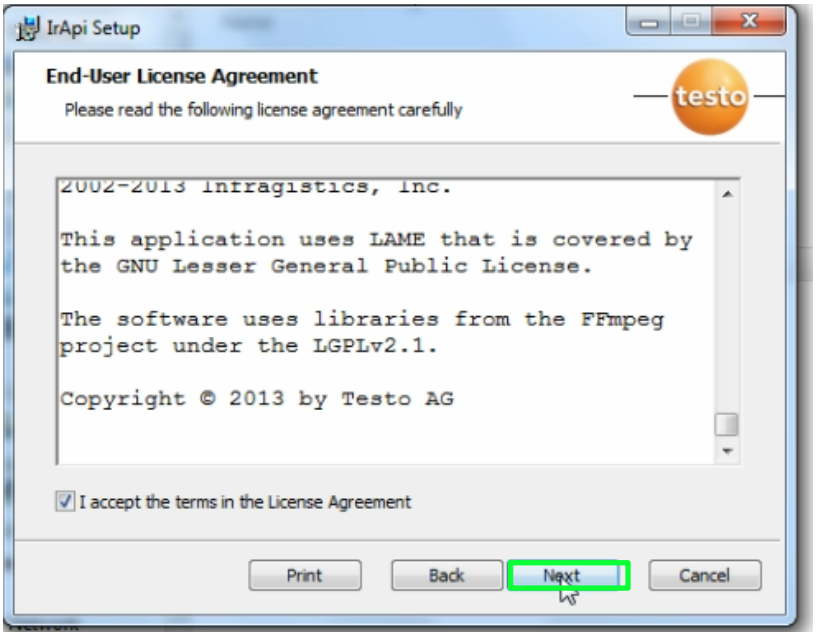


Please choose “Accept” and “Install” to continue the setup.

2.2 Main setup

The main setup installs the *IrApi* itself. The setup will prompt you a welcome dialog, where you continue by click “Next”.

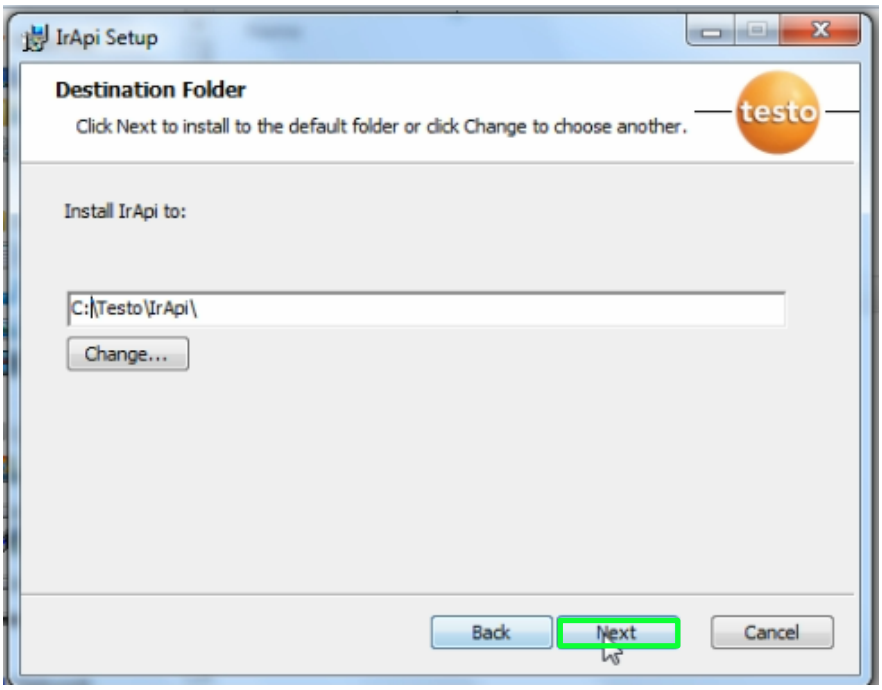
To continue you should accept the End-User License Agreement (EULA) and click next.



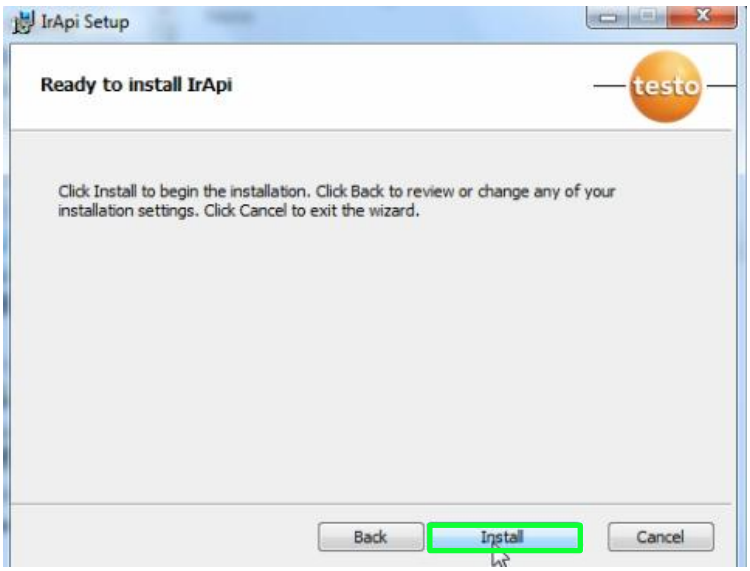
Now you can choose the path for the *IrApi* installation. Per default the *IrApi* would install in "%Programfiles%\Testo\IrApi" or under x64 "%Programfiles(x86)%\Testo\IrApi".

Be careful: you must not choose the Testo *IRSoft* path.

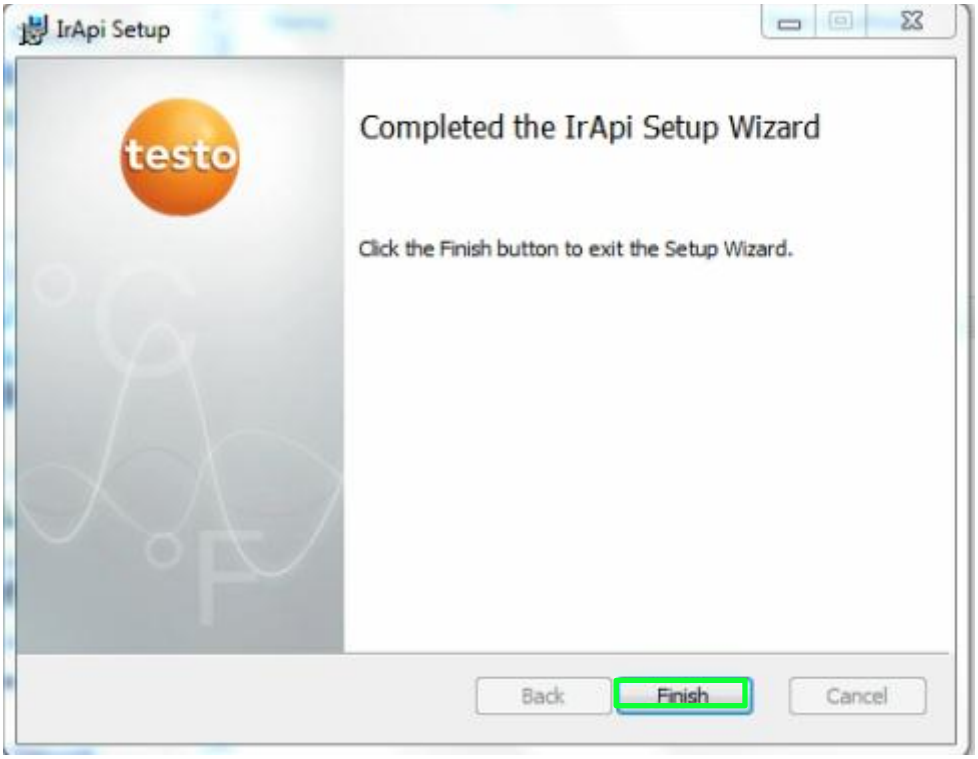
Click *Next* to continue.



As last step you must allow the installation to start. Therefore click "Install".



After the installation is finished, it will prompt you a summary.



2.3 Silent

It is **not** possible to start the whole setup as a silent setup so no user interaction is necessary. If you need a silent install, you can install the prerequisites and the IrApi.msi silently. Be careful that all prerequisites are installed before you install IrApi.msi.

2.4 Installed Components

Besides the files needed by the *IrApi* and this documentation you find a folder *examples* in the installation folder of *IrApi*. In this folder you can find a solution created with Visual Studio 2010 including one example usage with C# und one using plan C.

If you do not have the possibility to use Visual Studio 2010 please have a look at code in the subfolder of the example directory.

3 Getting Started

There are components which need to load in your application target directory. Otherwise the application will fail to call any *IrApi* functions.

3.1 Way to get it run

Please ensure that your application finds the following files originally located in the installation path of the *IRApi*:

opencv_calib3d242.dll
opencv_calib3d242d.dll
opencv_contrib242.dll
opencv_contrib242d.dll
opencv_core242.dll
opencv_core242d.dll
opencv_features2d242.dll
opencv_features2d242d.dll
opencv_flann242.dll
opencv_flann242d.dll
opencv_highgui242.dll
opencv_highgui242d.dll
opencv_imgproc242.dll
opencv_imgproc242d.dll
opencv_legacy242.dll
opencv_legacy242d.dll
opencv_ml242.dll
opencv_ml242d.dll
opencv_nonfree242.dll
opencv_nonfree242d.dll
opencv_objdetect242.dll
opencv_objdetect242d.dll
opencv_photo242.dll
opencv_photo242d.dll
opencv_stitching242.dll
opencv_stitching242d.dll
opencv_video242.dll
opencv_video242d.dll
opencv_videostab242.dll
opencv_videostab242d.dll

UsbLocID100.dll
wdapi1020.dll
zlib1.dll

t880_x.txt
t880_y.txt
t881_t875_x.txt
t881_t875_y.txt
t882_x.txt
t882_y.txt

TiLibAnalyzing.dll
TiLibCam.dll
TiLibCamComGen3.dll
TiLibCamComGen3Usb.dll
TiLibCodecs.dll
TiLibColor.dll
TiLibCommon.dll
TiLibCore.dll
TiLibDataTypes.dll
TiLibEnhancement.dll
TiLibFeatureDetection.dll
TiLibMarker.dll
TiLibMath.dll
TiLibRadiometry.dll
TiLibStitching.dll
TiLibStreaming.dll

ThermalImageApi.dll
Emgu.CV.dll
Emgu.Util.dll
Testo.Framework.Archive.BL.Interfaces.dll
Testo.Framework.Archive.DA.Interfaces.dll
Testo.Library.Measurement.dll
TestoIRArchive.Interfaces.dll
TestoIRBase.dll
TestoIRImage.dll

You can do that by editing the global path variable or by simply copying all the files to the folder your application is running in.

4 .NET

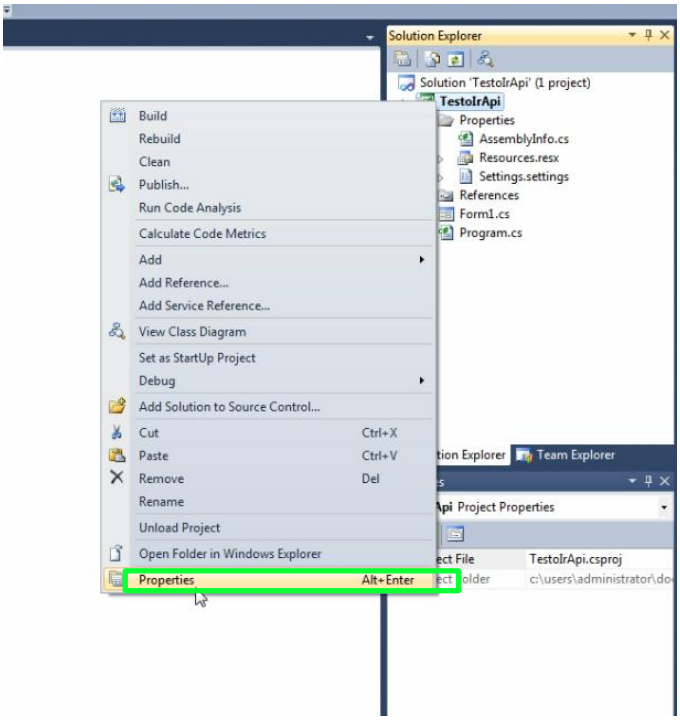
In the following section the usage of .NET interface is described.

The screenshots are made using Visual Studio .NET 2010 but of course you can use every build environment you want.

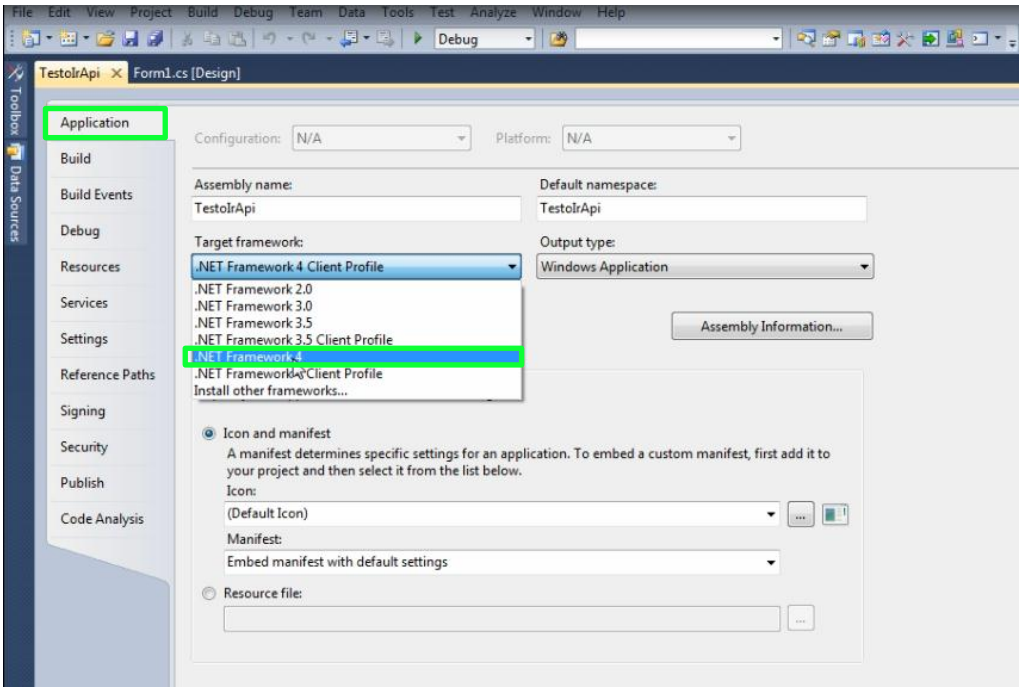
4.1 Project Settings

Please select ".Net Framework 4" as the target framework.

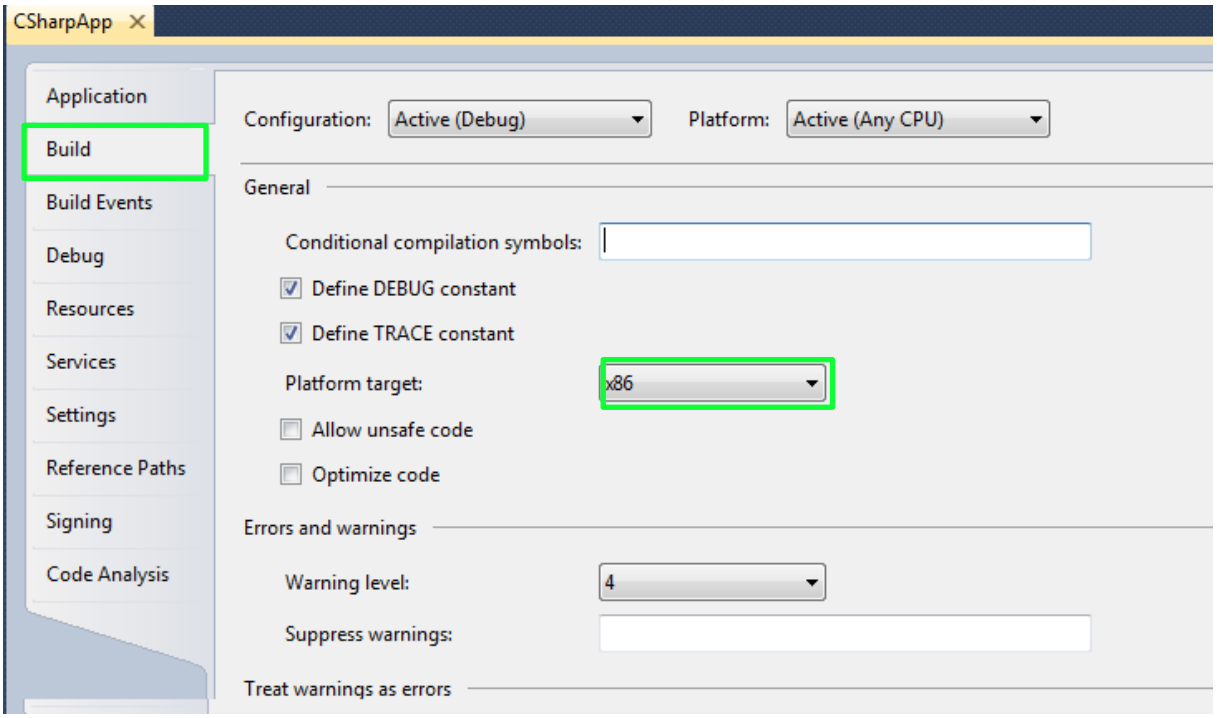
In Visual Studio 2010 you can do that by editing the properties of the solution.



Select Application and choose .NET Framework 4 as Target framework.



Switch to Build an set Platform target to x86.



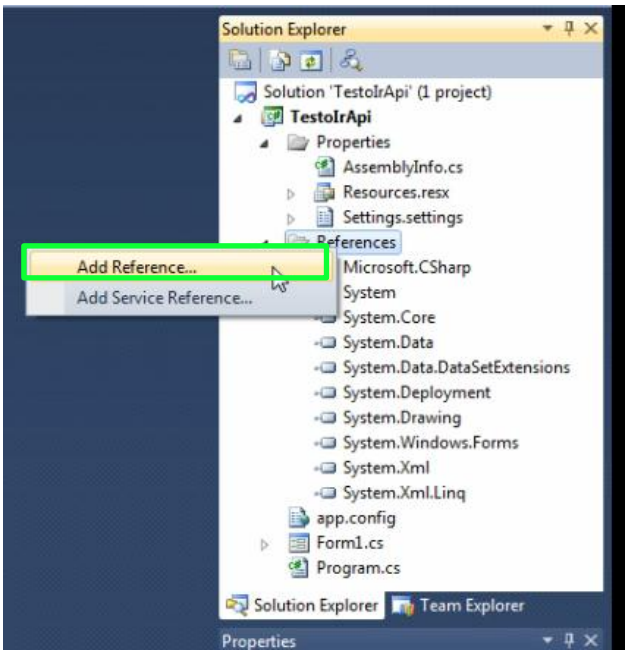
4.2 References

Please set references to the following 3 DLLs:

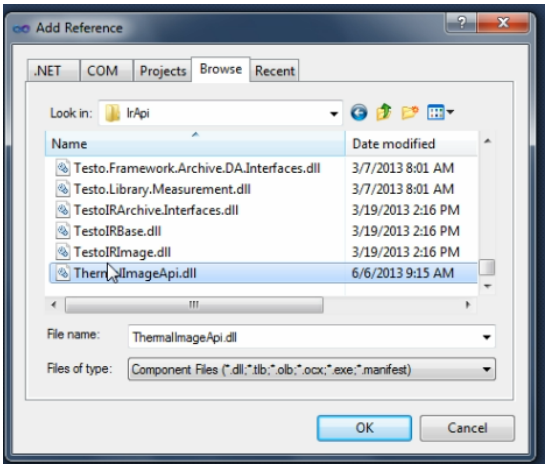
- ThermalImageApi.dll
- Testo.Library.Measurement.dll
- Testo.IRImage.dll

All of this files you can find in the installation folder of the *IRApi*.

In Visual Studio 2010 you can do this by chossing “AddReference…” of the context menu of the solution explorer.

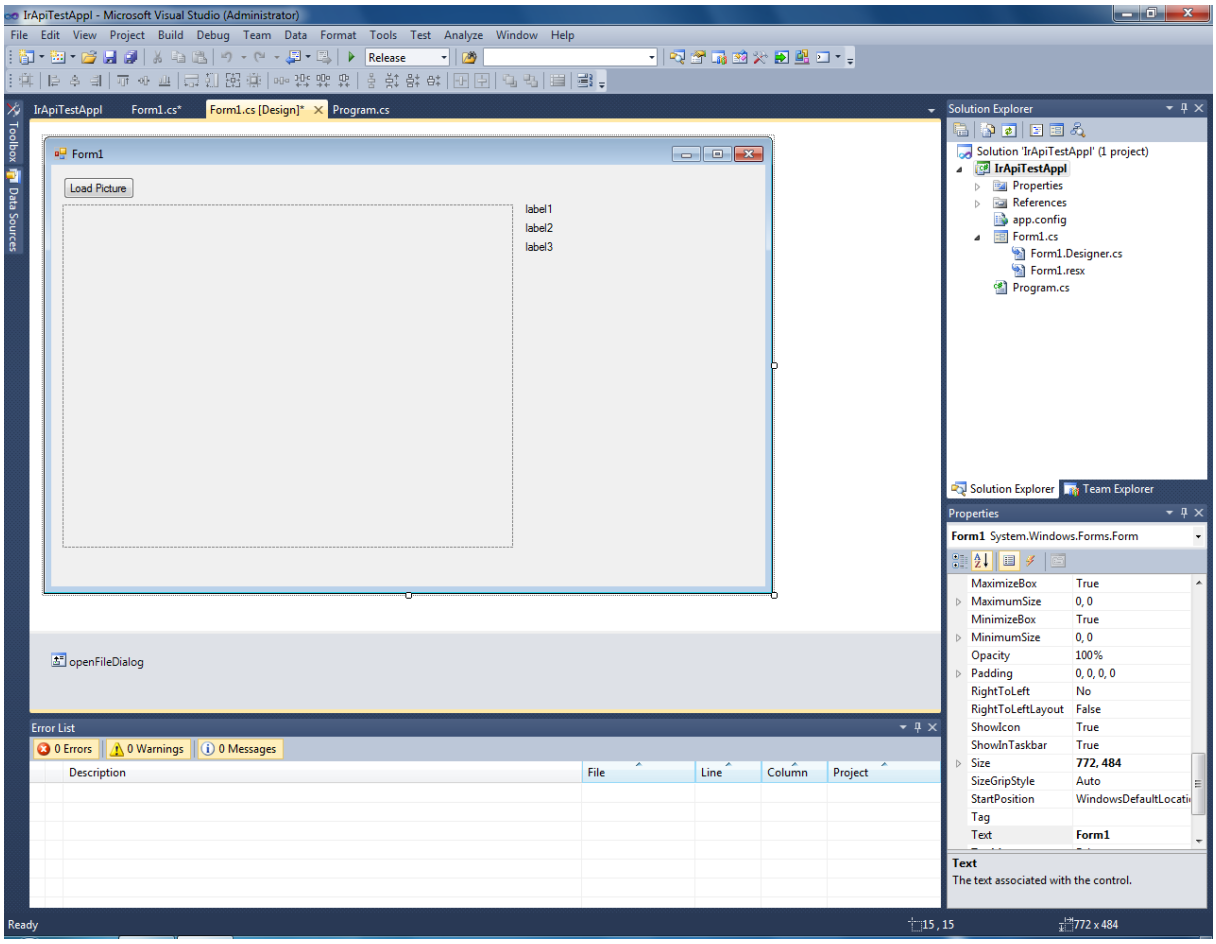


A new window pops up and you must select Browse. Browse to the installation folder of *IrApi*, mark the file and press ok.



4.3 Simple Example

Create a WinForms application like this.



Add using directives and instantiate the ThermalImageApi.dll.

```
ThermalImageApi tImage = new ThermalImageApi();
```

Now you can open picture using the following line.

```
tImage.Open(openFileDialog.FileName);
```

With the information in tImage you could read out e.g.: the device name, serial number or the Field of View (FoV).

```
deviceLabel.Text = "Device: "+tImage.DeviceName;
serialNumberLabel.Text = "Serial Number: " + tImage.SerialNumber;
foVLabel.Text = "FoV: " + tImage.FoV;
```

In the end you can show the image as thermal image in a picturebox. That could be done with following code.

```
pictureBox.Image = tImage.GetThermalImage(Unit.GradC);
```

The whole code could look like this:

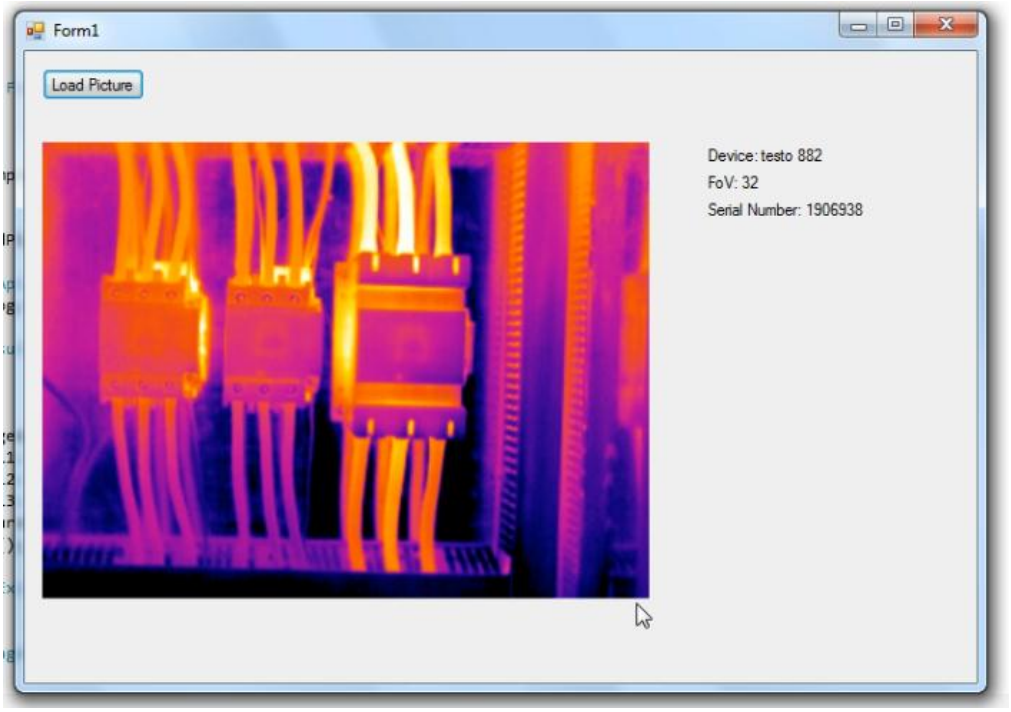
```
using System;
using System.Windows.Forms;
using Testo.IRSoft.API.Image;
using Testo.Library.Measurement;

namespace IrApiTestAppl
{
    public partial class Form1 : Form
    {
        public Form1()
        {
            InitializeComponent();
        }

        private void LoadPicButton_Click(object sender, EventArgs e)
        {
            openFileDialog.Filter = "bmt files (*.bmt)|*.bmt";
            if (DialogResult.OK == openFileDialog.ShowDialog())
            {
                try
                {
                    ThermalImageApi tImage = new ThermalImageApi();
                    tImage.Open(openFileDialog.FileName);
                    deviceLabel.Text = "Device: "+tImage.DeviceName;
                    serialNumberLabel.Text = "Serial Number: " + tImage.SerialNumber;
                    foVLabel.Text = "FoV: " + tImage.FoV;
                    pictureBox.Image = tImage.GetThermalImage(Unit.GradC);
                }
                catch (Exception ex)
                {
                    MessageBox.Show("Error: "+ ex.Message);
                }

            }
        }
    }
}
```

If you run this code, you should see something similar like this:



4.4 ThermalImageApi.dll

The ThermalImageApi.dll provides following methods and properties to work with *IrApi*.

4.4.1 Temperature Unit

All temperatures are in °C.

4.4.2 Exception handling

All internal exceptions of the *IrApi* are of the type IRApiImageException, which is derieved from System.Exception.

4.4.3 Construction

The ThermalImageApi has a standard constructor.

```
ThermalImageApi();
```

4.4.4 Methods

Name Open
Parameter `string path`
Return type: `void`
Description: Opens the file and loads the .bmt-image into the memory.

Name GetThermalImage
Parameter `Testo.Library.Measurement.Unit value`
Return type: `Bitmap`
Description: Gets a thermal image from .bmt-file.
Attention: use only the temperature units GradC or GradF.

Name GetThermalImageWithPalette
Parameter `Testo.Library.Measurement.Unit value`
Return type: `Bitmap`
Description: Gets a thermal bitmap with scale from .bmt-file. Attention: use only the temperature units GradC or GradF.

Name GetVisualImage
Parameter Non
Return type: `Bitmap`/NULL
Description: Gets the real image from .bmt-file if existing; null otherwise.

Name GetMeasurementRange
Parameter `ref float min, ref float max`
Return type: `void`
Description: Returns the measurement range information of the .bmp-fil.

Name GetTemperature
Parameter `int x, int y`
Return type: `float`
Description: Returns the temperature value of a point of the .bmt-file.
The point of origin is on the upper left corner.

4.4.4.1 Device information

Name DeviceName
Return type: string
Getter/Setter: Getter
Description: Returns the type of IR-Camera.
Possible Value: Any camera
e.g.: testo 882
t890-2

Name FoV (Field of View)
Return type: int
Getter/Setter: Getter
Description: Returns the view angle in angle degree.
E.g.: an angle of 42° indicates wide angle lens.

Name SerialNumber
Return type: uint
Getter/Setter: Getter
Description: Returns the serial number of IR-Camera.

4.4.4.2 Picture information

Name CreationDateTime
Return type: DateTime
Getter/Setter: Getter
Description: Returns the date and time of shooting the IR-Image. The date/time would written in the currently thread culture. See Microsoft for more information.

Name Height
Return type: int
Getter/Setter: Getter
Description: Returns the height of the picture in Pixel.

Name Width
Return type: int
Getter/Setter: Getter
Description: Returns the width of the picture in Pixel.

Name ReflectedTemperature
Return type: double
Getter/Setter: Getter and Setter
Description: Returns or sets the reflected temperature value for an IR-Image. This value has influence on the calculation of the temperatures.

Name Emissivity
Return type: double
Getter/Setter: Getter and Setter
Description: Returns or sets the emissivity value for an IR-Image. This value has influence on the calculation of the temperatures.

Name MinScaleTemperature
Return type: float
Getter/Setter: Getter and setter
Description: Returns or sets the minimal scale temperature value. This value stands for the lowest scale temperature.

Name MaxScaleTemperature
Return type: float
Getter/Setter: Getter and setter
Description: Returns or sets the maximal upper scale temperature value. This value stands for the highest scale temperature.

Name Palette
Return type: Testo.IRSoft.Image.Palette
Getter/Setter: Getter and Setter
Description: Returns or sets the color palette of the picture.
Possible Value: IronBow = 0,
RainBow = 1,
GreyScale = 2,
GreyScaleInv = 3,
Sepia = 4,
BlueRed = 5,
HotCold = 6,
Testo = 7,
DewPoint = 8,
Hochtemp = 9,
RainbowHC = 10

4.4.4.3 Humidity

Name Humidity
Return type: double
Getter/Setter: Getter and setter
Description: Returns or sets the humidity in %rH.

4.4.4.4 Highlighting

Name UseLimits
Return type: bool
Getter/Setter: Getter and setter
Description: Turns on or off if limits are used. Returns current state.

Name UseIsotherm
Return type: bool
Getter/Setter: Getter and setter
Description: Turn on or off if isotherm highlighting is used. Returns current state

Name LowerIsoTemperature
Return type: float
Getter/Setter: Getter and setter
Description: Returns or sets the lower isotherm temperature value. This value stands for the lowest highlighted isotherm temperature.

Name UpperIsoTemperature
Return type: float
Getter/Setter: Getter and setter
Description: Returns or sets the upper isotherm temperature value. This value stands for the highest highlighted isotherm temperature.

Name LowerLimitTemperature
Return type: float
Getter/Setter: Getter and setter
Description: Returns or sets the lower limit temperature value. This value stands for the lowest limit temperature.

Name UpperLimitTemperature
Return type: float
Getter/Setter: Getter and setter
Description: Returns or sets the upper limit temperature value. This value stands for the highest limit temperature.

5 C

In the following section the usage of the C interface is described.

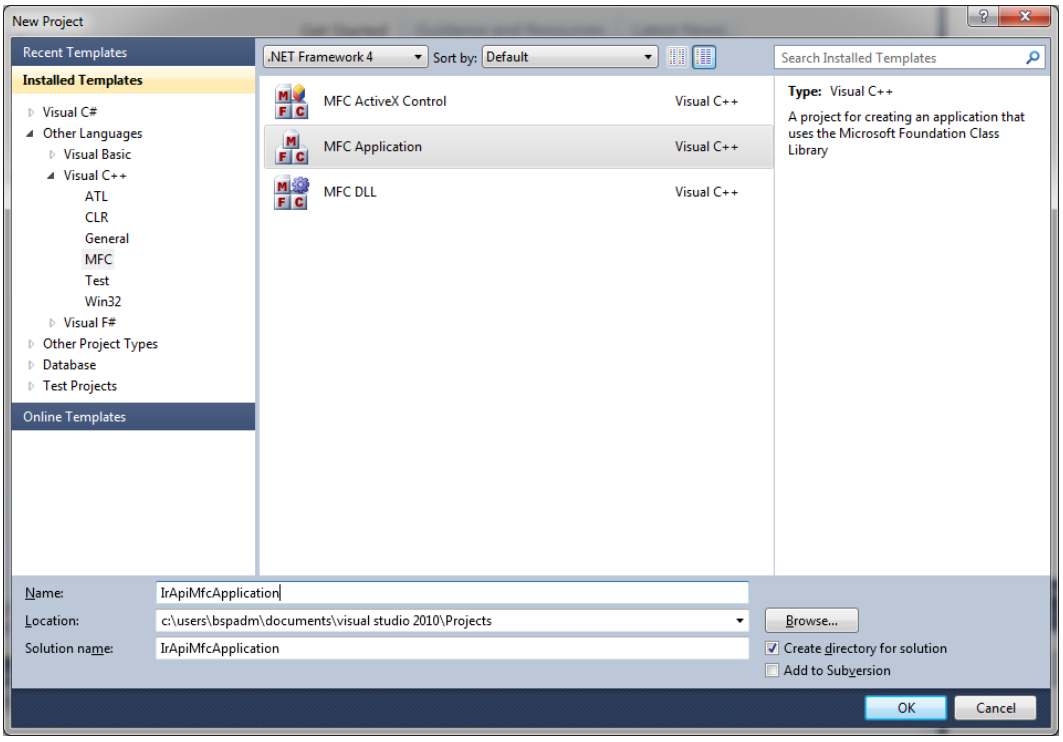
The screenshots are made using Visual Studio 2010 but of course you can use every build environment you want.

5.1 Getting started

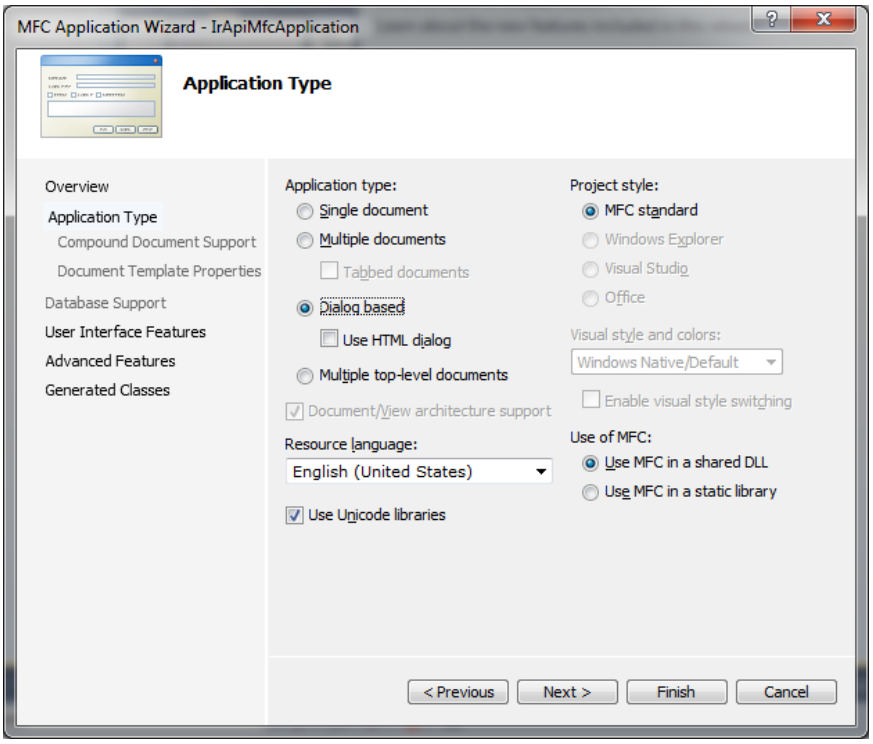
There are components which need to be loaded in your application target directory. Otherwise the application will fail to call any IrApi function. Make sure that all Files from chapter “3.1 Way to get it run” are in the target directory of your program.

5.2 Simple Example

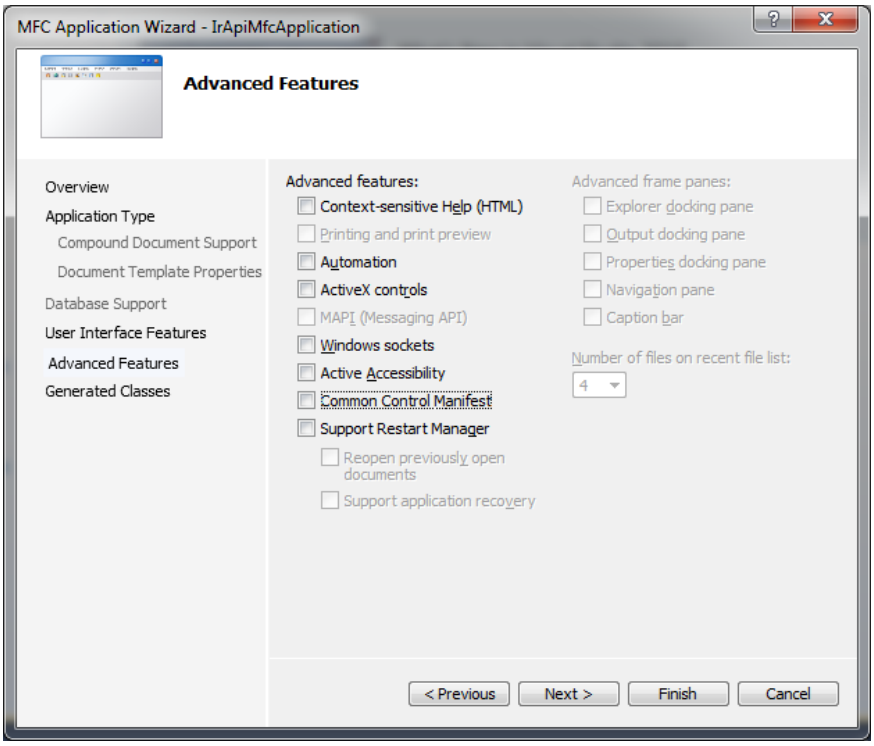
In this simple example you we use a MFC Dialog Application.



Choose “Dialog based” Application Type.

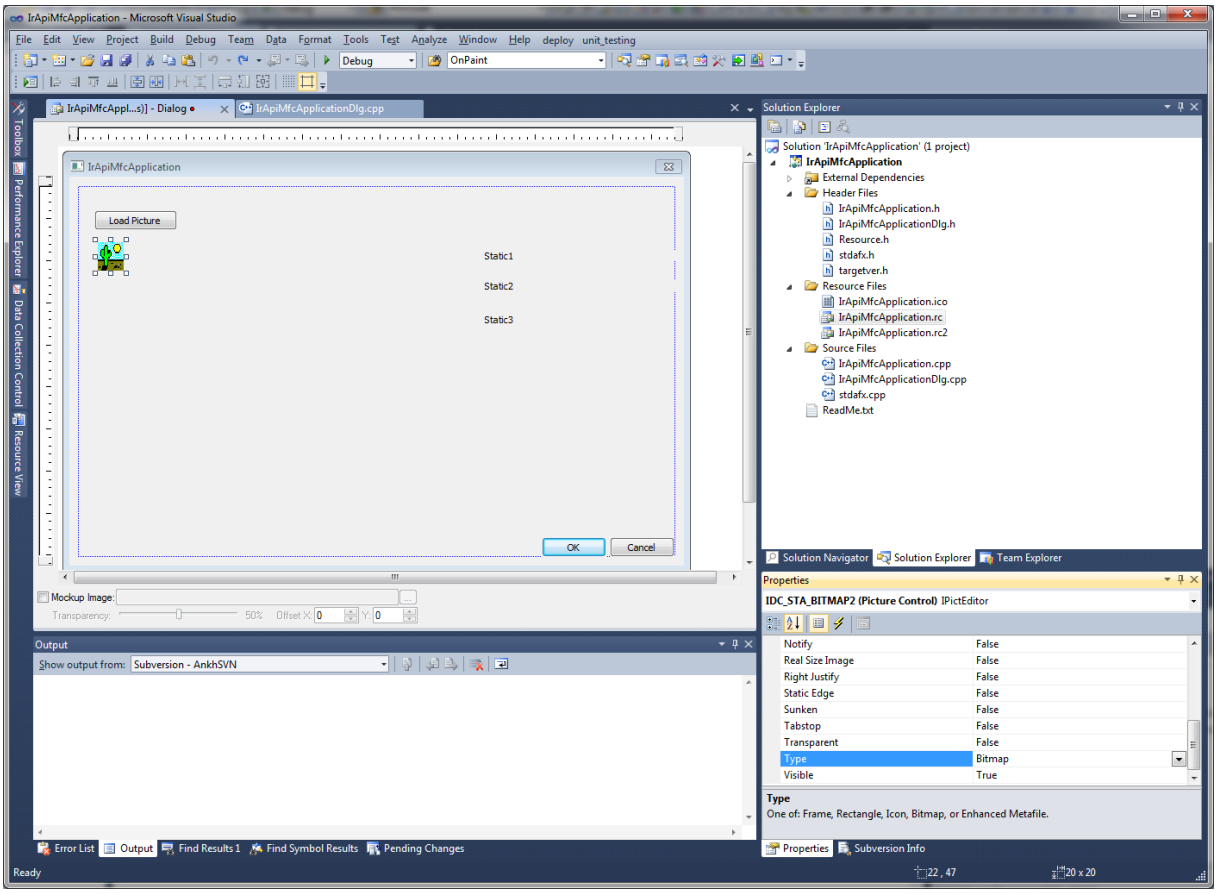


You can deselect advanced features.



You can “Finish” the Application Wizard.

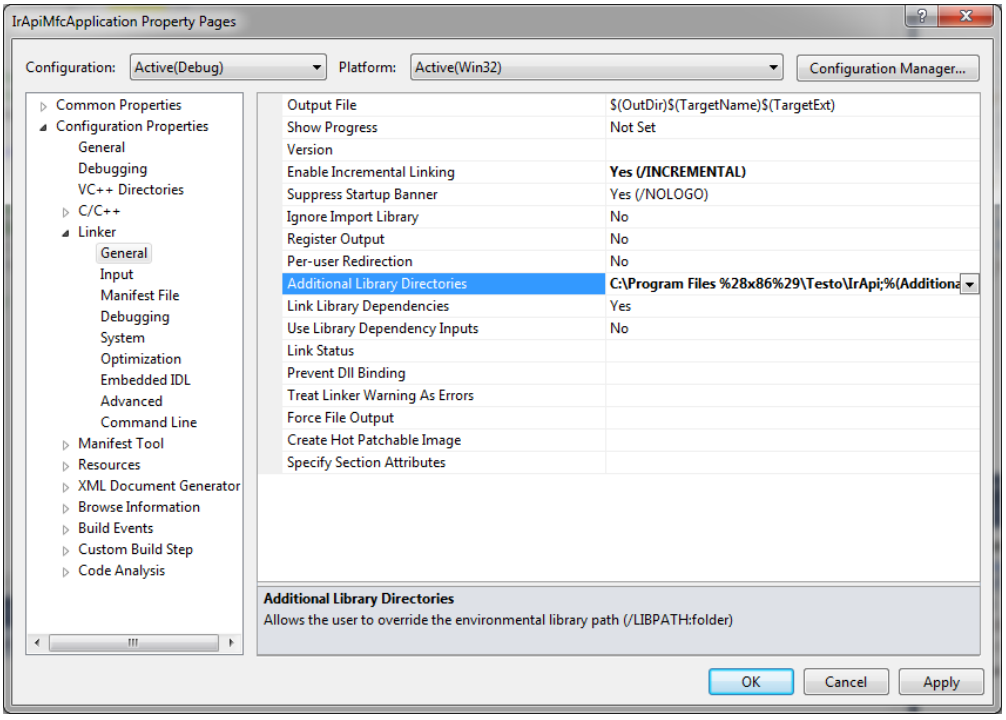
Place some Controls on your Dialog.



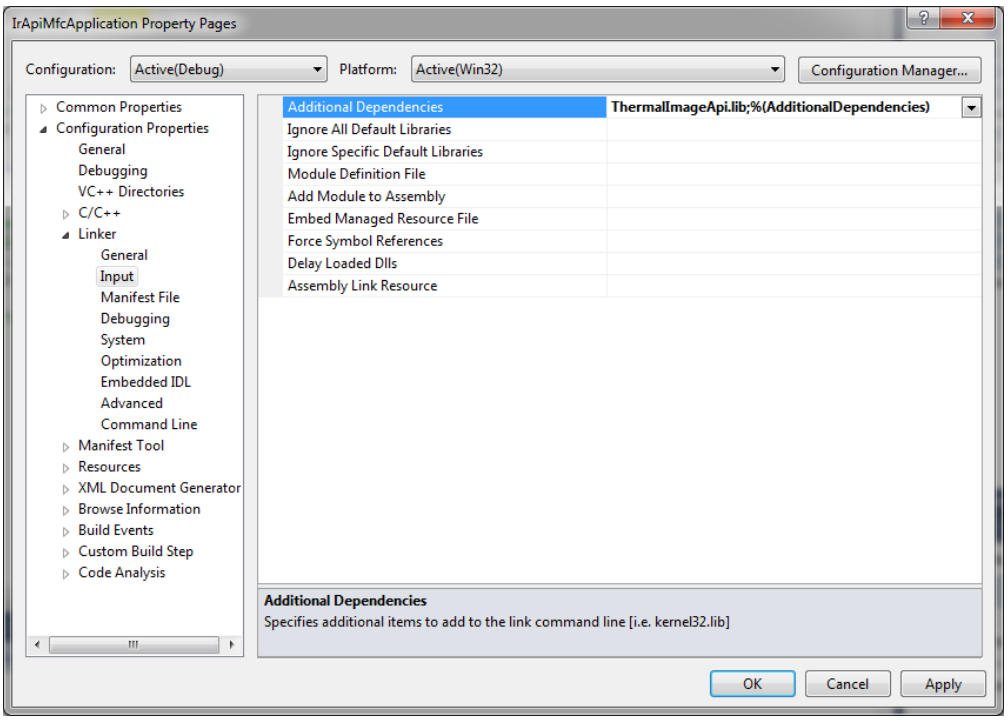
For the Picture Control you should use the “Bitmap” type.

To use the C interface you should include the ThermalImageApi.lib to your project. You can do that by editing the project properties.

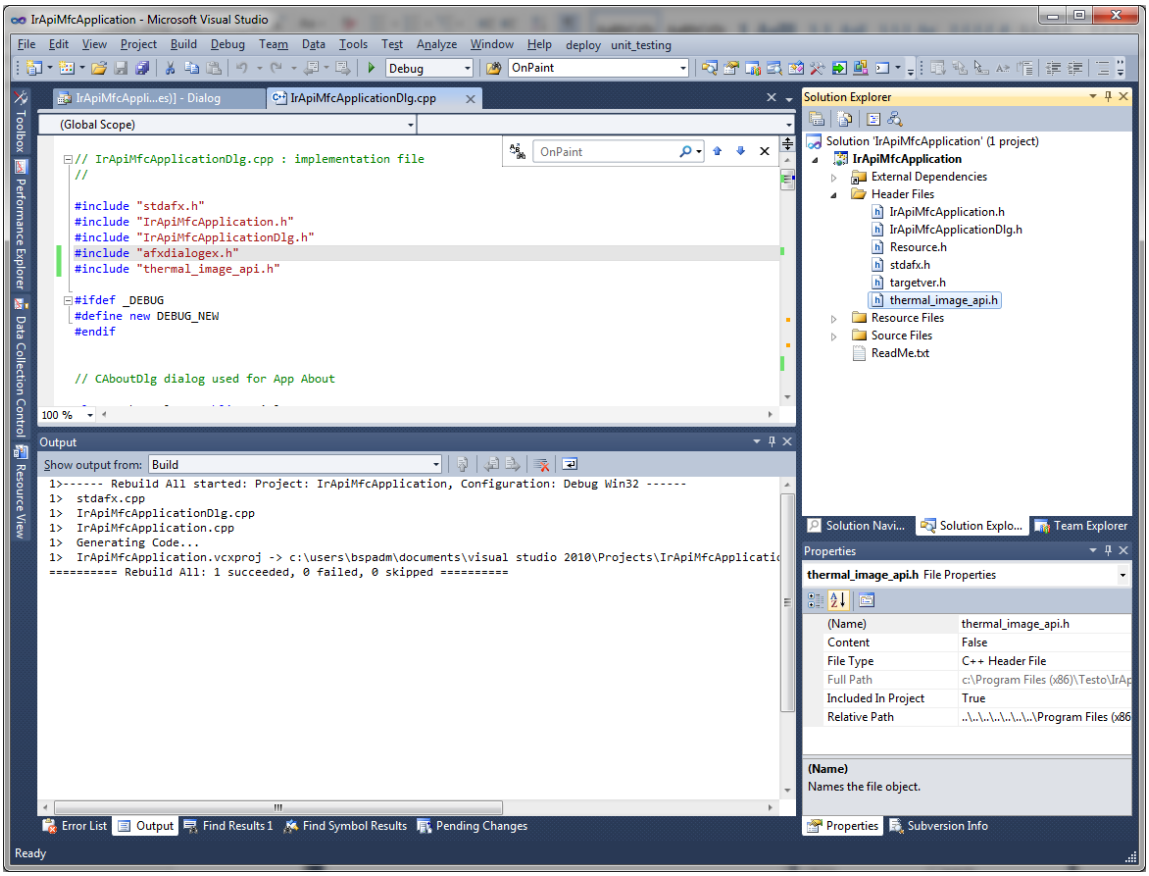
Add your installation folder of the IrApi as an “Additional Library Directory”.



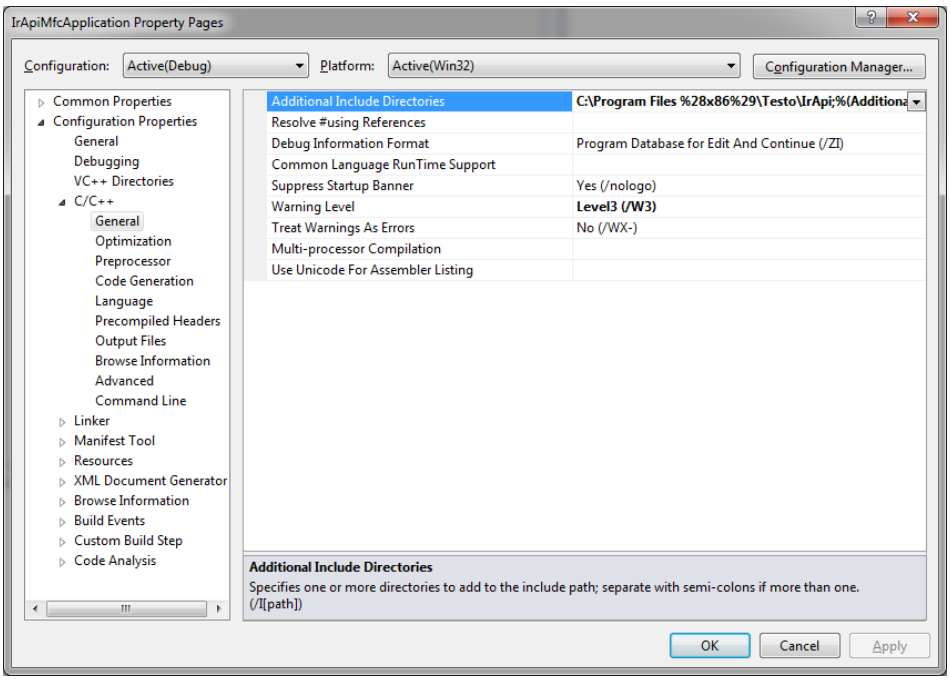
Add the ThermalImageApi.lib as an “Additional Dependency”.



For calling the IrApi functions you need to include the thermal_image_api.h (header) file.



You should add an “Additional Include Directory”



With this settings you have done so far, you can implement the “Load Picture” event handler.

```
// <summary>
/// Called when clicked on loadimage button.
/// </summary>
void CIrApiMfcApplicationDlg::OnBnClickedButLoadpicture()
{
   wchar_t resultBuffer[100];
   try
   {
      int id;
      USES_CONVERSION;
      LPCTSTR fileName = _T("*.bmt");
      CFileDialog fileDlg(TRUE/*Open=TRUE Save=False*/,
                     _T("bmt")/*Filename Extension*/,
                     fileName/*Initial Filename*/,
                     OFN_ENABLESIZING|OFN_EXPLORER|OFN_FILEMUSTEXIST/*Flags*/,
                     _T("*.bmt")/*Filetype Filter*/,
                     this/*parent Window*/);
      if (fileDlg.DoModal() == IDOK)
      {

         CString fileName = fileDlg.GetPathName();

         // get the ir image
         TESTO_IRAPI_RESULT tresult = testo_irimage_open(&id, (const wchar_t*)fileName);
         swprintf(resultBuffer, 100, L"Result: %s\n", testo_irimage_error_string(tresult));

         wchar_t path[_MAX_PATH];
         tresult = testo_irimage_thermal_image(id, TESTO_IRAPI_UNIT_CELSIUS, sizeof(path), path);
         swprintf(resultBuffer, 100, L"Result: %s\n", testo_irimage_error_string(tresult));
         HBITMAP startBitmap = (HBITMAP)LoadImage(NULL, path, IMAGE_BITMAP,400,300, LR_LOADFROMFILE);
         m_picture.SetBitmap(startBitmap);

         const int textLength = 100;
         wchar_t text_buffer[textLength+1];
         wchar_t display_text[textLength+1];

         // get the device name
         tresult = testo_irimage_get_devicename(id, textLength, text_buffer);
         swprintf(resultBuffer, 100, L"Result: %s\n", testo_irimage_error_string(tresult));
         swprintf(display_text, textLength, L"Device: %s\n", text_buffer);
         m_label1.SetWindowTextW(display_text);

          // get the serial number
         unsigned int serial;
         tresult = testo_irimage_get_serialnumber(id, &serial);
         swprintf(resultBuffer, 100, L"Result: %s\n", testo_irimage_error_string(tresult));
         swprintf(display_text, textLength, L"Serial Number: %d\n", serial);
         m_label2.SetWindowTextW(display_text);

         // get the FOV
         int fov;
         tresult = testo_irimage_get_fov(id, &fov);
         swprintf(resultBuffer, 100, L"Result: %s\n", testo_irimage_error_string(tresult));
         swprintf(display_text, textLength, L"FoV: %d\n", fov);
         m_label3.SetWindowTextW(display_text);

          testo_irimage_close(id);
      }
   }
   catch(...)
   {
      MessageBox(resultBuffer);
   }
}
```

In this handler, with the CFileDialog, you can choose a picture file (.bmt). Then this file is opened with `testo_irimage_open` and an image is created with `testo_irimage_thermal_image`. This image is then loaded and displayed in the picture control of the dialog. Additionally the device name, serial number and FOV are read and displayed in the static text controls on the right side of the picture.

If you run this code, you should see something like this:

5.3 ThermalImageApi.dll

5.3.1 Temperature Unit

All temperatures are in °C.

5.3.2 ID

While `testo_irimage_open IrApi creats an imageID. This imageID would needed from all methods.`

5.3.3 Return value

TESTO_IRAPI_OK would return if everything is okey, otherwise an errorcode would transmitted.

5.3.4 Methods

Name testo_irimage_open
Parameter non
Reference parameter: int* id, const wchar_t* fname
Description: Opens the file and loads the .bmt-image into the memory.

Name testo_irimage_close
Parameter int id
Description: Closes the .bmt-image and unloads it from the memory

Name testo_irimage_thermal_image
Parameter int id, TESTO_IRAPI_UNIT unit, int length
Reference parameter: wchar_t* path
Description: Gets a thermal image from .bmt-file.
Attention: use only the temperature units GradC or GradF.

Name testo_irimage_thermal_image_with_palette
Parameter int id, TESTO_IRAPI_UNIT unit, int length
Reference parameter: wchar_t* path
Description: Gets a thermal bitmap with scale from .bmt-file. Attention: use only the temperature units GradC or GradF.

Name testo_irimage_visual_image
Parameter int id, int length
Reference parameter: wchar_t* path
Description: Gets the real image from .bmt-file if existing.

Name testo_irimage_get_measurement_range
Parameter int id
Reference parameter: floate* min, float* max
Description: Writes the measurement range information of the .bmp-file into given parameters.

Name testo_irimage_get_temperature
Parameter int x, int y
Reference parameter: floate* temperature
Description: Writes the temperature value of a point of the .bmt-file into given parameter. The point of origin is on the upper left corner.

5.3.4.1 Device information

Name testo_irimage_get_devicename
Parameter: int id, int length
Reference parameter: wchar_t* text_buffer
Getter/Setter: Getter
Description: Writes the type of IR-Camera into given parameter.
Possible Value: Any camera
e.g.: testo 882
t890-2

Name testo_irimage_get_fov (Field of View)
Parameter: int id
Reference parameter: int* fov
Getter/Setter: Getter
Description: Writes the view angle in angle degree into given parameter.
E.g.: an angle of 42° indicates wide angle lens.

Name testo_irimage_get_serialnumber
Parameter: int id
Reference parameter: unsigned int* serial
Getter/Setter: Getter
Description: Writes the serial number of IR-Camera into given parameter..

5.3.4.2 Picture information

Name testo_irimage_get_datetime
Parameter: int id, int length
Reference parameter: int* year ,int* month ,int* day ,int* hour ,int* minute ,int* second
Getter/Setter: Getter
Description: Writes the date and time of shooting the IR-Image into given parameter.

Name testo_irimage_get_height
Parameter: int id
Reference parameter: int* height
Getter/Setter: Getter
Description: Writes the height of the picture in Pixel into given parameter.

Name testo_irimage_get_width
Parameter: int id
Reference parameter: int* width
Getter/Setter: Getter
Description: Writes the width of the picture in Pixel into given parameter.

Name testo_irimage_get_reflected_temperature
Parameter: int id
Reference parameter: double* refl_temperature
Getter/Setter: Getter
Description: Writes the reflected temperature value for an IR-Image into given parameter.
This value has influence on the calculation of the temperatures.

Name testo_irimage_set_reflected_temperature
Parameter: int id, double refl_temperature
Getter/Setter: Setter
Description: Sets the reflected temperature value for an IR-Image.
This value has influence on the calculation of the temperatures.

Name testo_irimage_get_emissivity
Parameter: int id
Reference parameter: double* emissivity
Getter/Setter: Getter
Description: Writes the emissivity value for an IR-Image into given parameter.
This value has influence on the calculation of the temperatures.

Name testo_irimage_set_emissivity
Parameter: double emissivityValue
Getter/Setter: Setter
Description: Sets the emissivity value for an IR-Image.
This value has influence on the calculation of the temperatures.

Name testo_irimage_get_min_scale
Parameter: int id
Reference parameter: float* minScale
Getter/Setter: Getter
Description: Writes the minimal scale temperature value into given parameter.
This value stands for the lowest scale temperature.

Name testo_irimage_set_min_scale
Parameter: int id, float minScale
Getter/Setter: Setter
Description: Sets the minimal scale temperature value. This value stands for the lowest scale temperature.

Name testo_irimage_get_max_scale
Parameter: int id
Reference parameter: float* maxScale
Getter/Setter: Getter
Description: Writes the maximal upper scale temperature value into given parameter. This value stands for the highest scale temperature.

Name testo_irimage_set_max_scale
Parameter: int id, float maxScale
Getter/Setter: Setter
Description: Sets the maximal upper scale temperature value. This value stands for the highest scale temperature.

Name testo_irimage_get_palette
Parameter: int id
Reference parameter: TESTO_IRAPI_PALETTE* Palette
Getter/Setter: Getter
Description: Writes the color palette of the picture into given parameter.
Possible Value: TESTO_IRAPI_PALETTE_IRONBOW =0,
TESTO_IRAPI_PALETTE_RAINBOW =1,
TESTO_IRAPI_PALETTE_GREYSCALE =2,
TESTO_IRAPI_PALETTE_GREYSCALEINV =3,
TESTO_IRAPI_PALETTE_SEPIA =4,
TESTO_IRAPI_PALETTE_BLUERED =5,
TESTO_IRAPI_PALETTE_HOTCOLD =6,
TESTO_IRAPI_PALETTE_TESTO =7,
TESTO_IRAPI_PALETTE_DEWPOINT =8,
TESTO_IRAPI_PALETTE_HOCHTEMP =9,
TESTO_IRAPI_PALETTE_RAINBOWHC=10

Name testo_irimage_set_palette
Parameter: int id
Reference parameter: TESTO_IRAPI_PALETTE* Palette
Getter/Setter: Setter
Description: Sets the color palette of the picture.
Possible Value: TESTO_IRAPI_PALETTE_IRONBOW =0,
TESTO_IRAPI_PALETTE_RAINBOW =1,
TESTO_IRAPI_PALETTE_GREYSCALE =2,
TESTO_IRAPI_PALETTE_GREYSCALEINV =3,
TESTO_IRAPI_PALETTE_SEPIA =4,
TESTO_IRAPI_PALETTE_BLUERED =5,
TESTO_IRAPI_PALETTE_HOTCOLD =6,
TESTO_IRAPI_PALETTE_TESTO =7,
TESTO_IRAPI_PALETTE_DEWPOINT =8,
TESTO_IRAPI_PALETTE_HOCHTEMP =9,
TESTO_IRAPI_PALETTE_RAINBOWHC=10

5.3.4.3 Humidity

Name testo_irimage_get_humidity
Parameter: int id
Reference parameter: double* humidity
Getter/Setter: Getter
Description: Writes the humidity in %rH into given parameter.

Name testo_irimage_set_humidity
Parameter: int id, double humidity
Getter/Setter: Setter
Description: Sets the humidity in %rH.

5.3.4.4 Highlighting

Name testo_irimage_limits_applied
Parameter: int id
Reference parameter: int* uselimits
Getter/Setter: Getter
Description: Writes current state into given parameter.

Name testo_irimage_apply_limits
Parameter: int id, int uselimits
Getter/Setter: Setter
Description: Turns on or off if limits are used.

Name testo_irimage_iso_applied
Parameter: int id
Reference parameter: int* useIsotherme
Getter/Setter: Getter
Description: Write current state into given parameter.

Name testo_irimage_apply_iso
Parameter: int id, int useIsotherme
Getter/Setter: Setter
Description: Turn on or off if isotherm highlighting is used.

Name testo_irimage_get_lower_iso_temperature
Parameter: int id
Reference parameter: float* lowerIsoTemp
Getter/Setter: Getter
Description: Writes the lower isotherm temperature value into given parameter. This value stands for the lowest highlighted isotherm temperature.

Name testo_irimage_get_lower_iso_temperature
Parameter: int id, float lowerIsoTemp
Getter/Setter: Setter
Description: Sets the lower isotherm temperature value. This value stands for the lowest highlighted isotherm temperature.

Name testo_irimage_get_upper_iso_temperature
Parameter: int id
Reference parameter: float* upperIsoTemp
Getter/Setter: Getter
Description: Writes the upper isotherm temperature value into given parameter. This value stands for the highest highlighted isotherm temperature.

Name testo_irimage_set_upper_iso_temperature
Parameter: int id, float upperIsoTemp
Getter/Setter: Setter
Description: Sets the upper isotherm temperature value. This value stands for the highest highlighted isotherm temperature.

Name testo_irimage_get_lower_limit_temperature
Parameter: int id
Reference parameter: float* lowerLimitTemp
Getter/Setter: Getter
Description: Writes the lower limit temperature value into given parameter. This value stands for the lowest limit temperature.

Name testo_irimage_set_lower_limit_temperature
Parameter: int id, float lowerLimitTemp
Getter/Setter: Setter
Description: Sets the lower limit temperature value. This value stands for the lowest limit temperature.

Name testo_irimage_get_upper_limit_temperature
Parameter: int id
Reference parameter: float* upperLimitTemp
Getter/Setter: Getter
Description: Writes the upper limit temperature value into given parameter. This value stands for the highest limit temperature.

Name testo_irimage_set_upper_limit_temperature
Parameter: int id, float upperLimitTemp
Getter/Setter: Setter
Description: Sets the upper limit temperature value. This value stands for the highest limit temperature.