https://www.ximea.com/support/wiki/apis/XiAPINET

xiAPI.NET

xiAPI.NET は、XIMEA Application Programming Interface for Dot Net - **Microsoft Visual C#** の略です。
これは、すべての XIMEA カメラの新しい共通インターフェイスであり、汎用 xiAPI の簡易バージョンを表します。

アーキテクチャ

API a は、カメラ、オペレーティング システム ドライバー、およびアプリケーション間のソフトウェア インターフェイスです。
xiAPI.NET ランタイムは、32 ビット Windows オペレーティング システムで *xiapi32.dll* を呼び出す *xiAPI.NET.dll* と、64 ビット Windows オペレーティング システムで *xiapi64.dll* を呼び出す *xiAPI.NETX64.dll* に実装された xiCam クラスに基づいています。

インストール

xiAPI.NET サポートを追加するには:
・XIMEA Windows ソフトウェア パッケージをインストールします
・コンポーネントを選択するときは、API と例がチェックされていることを確認します。
・アプリケーションには、利用可能なバージョンの .NET SDK のいずれかを使用してください。サポートされているバージョンは次のとおりです。
  ・Microsoft .NET Framework 4.7.2
  ・.NET Core 2.2
  ・.NET Core 3.1
・アプリケーション開発には、Microsoft Visual Studio 2013 以降のバージョンを使用することをお勧めします。

サンプル

.NET サンプル コード - さまざまな .NET の例については、このセクションを参照してください。
XIMEA Windows ソフトウェア パッケージをインストールすると、サンプル プロジェクトがディレクトリ XIMEA/Examples/xiAPI.NET にインストールされます。
カメラから 10 枚の画像をキャプチャし、それらを .bmp ファイルとして保存するサンプル コードを以下に示します。
露光時間は、 2 ミリ秒に設定され、ゲインは 5 dB に設定されます。

```
// Sample for XIMEA API.NET

using System;
using System.Collections.Generic;
using System.Text;
using System.Collections;
using System.Drawing;
using xiApi.NET;
namespace xiApi.NET_example
{
class Program
{
  static void Main(string[] args)
  {
    xiCam myCam = new xiCam();

    try
    {
      // Initialize first camera
      myCam.OpenDevice(0);

      // Set device exposure to 2 milliseconds
      int exposure_us = 2000;
      myCam.SetParam(PRM.EXPOSURE, exposure_us);

      // Set device gain to 5 decibels
      float gain_db = 5;
      myCam.SetParam(PRM.GAIN, gain_db);

      // Set image output format to monochrome 8 bit
      myCam.SetParam(PRM.IMAGE_DATA_FORMAT, IMG_FORMAT.MONO8);

      //Start acquisition
      myCam.StartAcquisition();
```

```
      // Capture images
      Bitmap myImage;
      int timeout = 1000;
      for (int i = 0; i < 10; i++)
      {
        myCam.GetImage(out myImage, timeout );
        string fName = string.Format("image{0}.bmp", i);
        myImage.Save(fName);
      }
      // Stop acquisition
      myCam.StopAcquisition();
    }

    catch (System.ApplicationException appExc)
    {
      // Show handled error
      Console.WriteLine(appExc.Message);
      System.Console.ReadLine();
      myCam.CloseDevice();
    }

    finally
    {
      myCam.CloseDevice();
    }
  }
} // end of Program
} // end of namespace xiApi.NET_example
```

インターフェース

xiAPI.NET の中核となるのは次の関数で、カメラ機能のほとんどを制御できます。

```
// open interface
xiCam.OpenDevice(int DevId);

// set parameter
myCam.SetParam(PRM paramName, int paramVal);
myCam.SetParam(PRM paramName, float paramVal);

// get parameter
myCam.GetParam(PRM paramName, out int paramVal);
myCam.GetParam(PRM paramName, out float paramVal);
myCam.GetParam(PRM paramName, out string paramVal);

// get next image from buffer
myCam.GetImage(out myImage);

// close interface
xiCam.CloseDevice();
```

オフライン処理

xiAPI.NET では、オフライン処理を使用して、すでにキャプチャされ保存された画像を処理できます。

詳細については、xiAPI.NET オフライン処理をご覧ください。

ドキュメント

xiAPI.NET マニュアル
.NET ビットマップで xiAPI.NET ピクセル データにアクセスする

https://www.ximea.com/support/wiki/apis/XiAPINET

xiAPI.NET

xiAPI.NET stands for XIMEA Application Programming Interface for Dot Net - **Microsoft Visual C#**
It is a new common interface for all XIMEA cameras and represents a simplified version of generic **xiAPI.**

Architecture

API a is the software interface between the camera, operating system driver and application.
xiAPI.NET run-time is based on xiCam class implemented in *xiAPI.NET.dll* calling the *xiapi32.dll* on 32bit Windows operating systems and *xiAPI.NETX64.dll* calling the *xiapi64.dll* on 64bit Windows operating systems.

Installation

To add xiAPI.NET support:
・install **XIMEA Windows Software Package**
・When choosing components make sure: API and Examples are checked.
・Use one of the available versions of the .NET SDK for your application, supported versions are:
  ・Microsoft .NET Framework 4.7.2
  ・.NET Core 2.2
  ・.NET Core 3.1
・We recommend to use Microsoft Visual Studio 2013 or a later version for application development.

Sample

.NET Sample code - See this section with various .NET examples.

After installation of the XIMEA Windows Software Package - the sample project is installed in the directory XIMEA/Examples/xiAPI.NET.

Here follows a sample code that captures ten images from the camera and saves them as .bmp files.

Exposure time is set 2 ms and gain is set to 5 dB.

```
// Sample for XIMEA API.NET

using System;
using System.Collections.Generic;
using System.Text;
using System.Collections;
using System.Drawing;
using xiApi.NET;
namespace xiApi.NET_example
{
class Program
{
  static void Main(string[] args)
  {
    xiCam myCam = new xiCam();

    try
    {
      // Initialize first camera
      myCam.OpenDevice(0);

      // Set device exposure to 2 milliseconds
      int exposure_us = 2000;
      myCam.SetParam(PRM.EXPOSURE, exposure_us);

      // Set device gain to 5 decibels
      float gain_db = 5;
      myCam.SetParam(PRM.GAIN, gain_db);

      // Set image output format to monochrome 8 bit
      myCam.SetParam(PRM.IMAGE_DATA_FORMAT, IMG_FORMAT.MONO8);

      //Start acquisition
      myCam.StartAcquisition();
```

```
      // Capture images
      Bitmap myImage;
      int timeout = 1000;
      for (int i = 0; i < 10; i++)
      {
        myCam.GetImage(out myImage, timeout );
        string fName = string.Format("image{0}.bmp", i);
        myImage.Save(fName);
      }
      // Stop acquisition
      myCam.StopAcquisition();
    }

    catch (System.ApplicationException appExc)
    {
      // Show handled error
      Console.WriteLine(appExc.Message);
      System.Console.ReadLine();
      myCam.CloseDevice();
    }

    finally
    {
      myCam.CloseDevice();
    }
  }
} // end of Program
} // end of namespace xiApi.NET_example
```

Interface

The core of xiAPI.NET are the following functions, which allow the control of most of the camera functionality.

```
// open interface
xiCam.OpenDevice(int DevId);

// set parameter
myCam.SetParam(PRM paramName, int paramVal);
myCam.SetParam(PRM paramName, float paramVal);

// get parameter
myCam.GetParam(PRM paramName, out int paramVal);
myCam.GetParam(PRM paramName, out float paramVal);
myCam.GetParam(PRM paramName, out string paramVal);

// get next image from buffer
myCam.GetImage(out myImage);

// close interface
xiCam.CloseDevice();
```

Offline Processing

xiAPI.NET allows to process of already captured and stored images using Offline Processing.
Please read more at **xiAPI.NET Offline Processing.**

Documentation

xiAPI.NET Manual
Accessing xiAPI.NET pixel data in .NET Bitmap