ADVACAM
Imaging the Unseen

C-sharp commandline examples

Online version: https://wiki.advacam.cz/wiki/C-sharp_commandline_examples

Contents

Dummy C# commandline example

```
using System;
using System.Runtime.InteropServices;

namespace ConsoleApp1 {
    class Program {
        [DllImport("pxcore.dll", CallingConvention = CallingConvention.Cdecl)]
        //public static extern int pxcInitialize(Int32 a, UInt64 b); // before 1.8.5
        //PXCAPI int pxcInitialize(const char *iniFile="pixet.ini");
        public static extern int pxcInitialize(string iniFile = "pixet.ini");

        [DllImport("pxcore.dll", CallingConvention = CallingConvention.Cdecl)]
        public static extern int pxcExit();

        [DllImport("pxcore.dll", CallingConvention = CallingConvention.Cdecl)]
        public static extern int pxcGetDevicesCount();
        static void Main(string[] args) {
            int rc;

            Console.WriteLine("pxcInitialize ...");
            //rc = pxcInitialize(0, 0); // before 1.8.5
            rc = pxcInitialize();
            Console.WriteLine($"rc={rc:D} (0 is OK)", rc);

            Console.WriteLine("pxcGetDevicesCount...");
            rc = pxcGetDevicesCount();
            Console.WriteLine("rc={0:D}\n", rc);

            if (rc > 0) {
                // do something now
            } else {
                Console.WriteLine("No devices detected\n");
            }

            Console.WriteLine("pxcExit...");
            rc = pxcExit();
            Console.WriteLine("rc={0:D} Press any key to exit", rc);

            Console.ReadKey();
        }
    }
}
```

Some more complex imports for inspiration

```
 [DllImport("pxcore.dll", CallingConvention = CallingConvention.Cdecl)]
 public static extern int pxcGetDeviceChipID(UInt32 deviceIndex, UInt32 chipIndex,
StringBuilder chipIDBuffer, UInt32 size);

 [DllImport("pxcore.dll", CallingConvention = CallingConvention.Cdecl)]
 public static extern int pxcGetDeviceDimensions(UInt32 deviceIndex, ref UInt32 width, ref
UInt32 height);

 [DllImport("pxcore.dll", CallingConvention = CallingConvention.Cdecl)]
 public static extern int pxcMeasureSingleFrameTpx3(UInt32 deviceIndex, double frameTime,
[Out] double[] frameToaITot, [Out] UInt16[] frameTotEvent, ref UInt32 size, UInt32 trgStg =
0);
```

C# data-driven commandline example

```
using System;
using System.Runtime.InteropServices;
using System.Text;

namespace ConsoleApp1 {
    class Program {
        [DllImport("pxcore.dll", CallingConvention = CallingConvention.Cdecl)]
        //public static extern int pxcInitialize(Int32 a, UInt64 b); // before 1.8.5
        //PXCAPI int pxcInitialize(const char *iniFile="pixet.ini");
        public static extern int pxcInitialize(string iniFile = "pixet.ini");

        [DllImport("pxcore.dll", CallingConvention = CallingConvention.Cdecl)]
        public static extern int pxcExit();

        [DllImport("pxcore.dll", CallingConvention = CallingConvention.Cdecl)]
        public static extern int pxcGetDevicesCount();

        //PXCAPI int pxcGetLastError(char* errorMsgBuffer, unsigned size);
        [DllImport("pxcore.dll", CallingConvention = CallingConvention.Cdecl)]
        public static extern int pxcGetLastError(StringBuilder errorMsgBuffer, UInt32
size);

        //PXCAPI int pxcGetDeviceName(unsigned deviceIndex, char* nameBuffer, unsigned
size);
```

```
        [DllImport("pxcore.dll", CallingConvention = CallingConvention.Cdecl)]
        public static extern int pxcGetDeviceName(UInt32 deviceIndex, StringBuilder
nameBuffer, UInt32 size);

        //PXCAPI int pxcMeasureTpx3DataDrivenMode(unsigned deviceIndex, double measTime,
const char* fileName, unsigned trgStg = PXC_TRG_NO, AcqEventFunc callback = 0, intptr_t
userData = 0);
        [DllImport("pxcore.dll", CallingConvention = CallingConvention.Cdecl)]
        public static extern int pxcMeasureTpx3DataDrivenMode(UInt32 deviceIndex, double
measTime, string fileName, UInt32 trgStg = 0, IntPtr callback = default, IntPtr userData =
default);
        //PXCAPI int pxcGetMeasuredTpx3PixelsCount(unsigned deviceIndex, unsigned*
pixelCount);
        [DllImport("pxcore.dll", CallingConvention = CallingConvention.Cdecl)]
        public static extern int pxcGetMeasuredTpx3PixelsCount(UInt32 deviceIndex, out
UInt32 pixelCount);

        // typedef struct _Tpx3Pixel {double toa; float tot; unsigned int index;}
Tpx3Pixel;
        [StructLayout(LayoutKind.Sequential)]
        public struct Tpx3Pixel
        {
            public double toa;
            public float tot;
            public UInt32 index;
        };

        //PXCAPI int pxcGetMeasuredTpx3Pixels(unsigned deviceIndex, Tpx3Pixel* pixels,
unsigned pixelCount);
        [DllImport("pxcore.dll", CallingConvention = CallingConvention.Cdecl)]
        public static extern int pxcGetMeasuredTpx3Pixels(UInt32 deviceIndex, [Out]
Tpx3Pixel[] pixels, UInt32 pixelCount);

        const UInt32 pixelBuffSize = 1000000;
        static Tpx3Pixel[] pixelBuffer = new Tpx3Pixel[pixelBuffSize];

        //typedef void (*AcqEventFunc)(intptr_t eventData, intptr_t userData);
        public delegate void AcqEventFunc(IntPtr eventData, IntPtr userData);
        public static void AcqEventCallback(IntPtr eventData, IntPtr userData)
        {
            Console.WriteLine("*** AcqEventCallback evd={0} ud={1}", eventData,
userData.ToInt64());
            UInt32 pixelCount;
            int rc = pxcGetMeasuredTpx3PixelsCount(0, out pixelCount);
```

```
            Console.WriteLine("    pxcGetMeasuredTpx3PixelsCount rc={0:D} pixelCount={1}
{2}", rc, pixelCount, pixelCount>pixelBuffSize ? "(data truncated)" : "");
            if (pixelCount > 0)
            {
                rc = pxcGetMeasuredTpx3Pixels(0, pixelBuffer, (pixelCount<pixelBuffSize) ?
pixelCount : pixelBuffSize);
                Console.WriteLine("    pxcGetMeasuredTpx3Pixels rc={0:D}", rc);
                for (int i = 0; i < Math.Min(10, pixelCount); i++)
                {
                    Console.WriteLine("        i:{0} pxIdx={1} toa={2} tot={3}", i,
pixelBuffer[i].index, pixelBuffer[i].toa, pixelBuffer[i].tot);
                }
                if (pixelCount>10) Console.WriteLine("        ...");
            }
        }

        public static string GetErrorMessage(bool isErr)
        {
            if (isErr)
            {
                StringBuilder errBuff = new StringBuilder(256);
                pxcGetLastError(errBuff, 256);
                return "err:'" + errBuff.ToString() + "'";
            }
            else return "OK";
        }

        static void Main(string[] args) { //
================================================================================
            int rc;
            StringBuilder errBuff = new StringBuilder(256);

            Console.WriteLine("pxcInitialize...");
            //rc = pxcInitialize(0, 0); // before 1.8.5
            rc = pxcInitialize();
            Console.WriteLine($"rc={rc:D} {GetErrorMessage(rc < 0)}");

            Console.WriteLine("pxcGetDevicesCount...");
            int devCnt = pxcGetDevicesCount();
            Console.WriteLine($"rc={devCnt:D} {GetErrorMessage(devCnt < 0)}");

            StringBuilder nameBuffer = new StringBuilder(256);
            Console.WriteLine("pxcGetDeviceName...");
            for (uint n = 0; n < devCnt; n++)
```

```
            {
                rc = pxcGetDeviceName(n, nameBuffer, 256);
                Console.WriteLine("    n:{0} rc={1:D} nam:{2}\n", n, rc,
nameBuffer.ToString());
            }
            if (devCnt > 0) {
                Console.WriteLine("Note: A data measured immediatelly afther init can
contain chip starting artifacts");
                Console.WriteLine("      In real application add wait user response, feve
seconds, dummy measurement or do sensor refresh");

                Console.WriteLine("pxcMeasureTpx3DataDrivenMode...");
                rc = pxcMeasureTpx3DataDrivenMode(0, 10.0, "", 0,
Marshal.GetFunctionPointerForDelegate(new AcqEventFunc(AcqEventCallback)), IntPtr.Zero);
                Console.WriteLine($"pxcMeasureTpx3DataDrivenMode end: rc={rc:D}
{GetErrorMessage(rc < 0)}");
            } else {
                Console.WriteLine("No devices detected\n");
            }

            Console.WriteLine("pxcExit...");
            rc = pxcExit();
            Console.WriteLine("rc={0:D}");
        }
    }
}
```

Related

- Binary core API
- Frame-based basic measurement flow
- Data-driven basic measurement flow