ENSAMBLADO .NET

CIDARMT

Manual de referencia

VERSIÓN 3.4

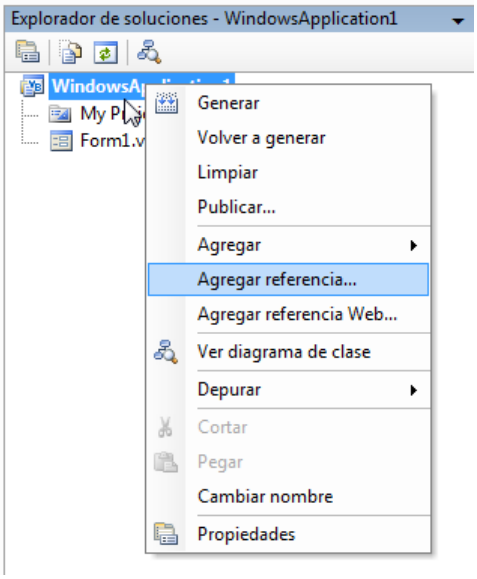
septiembre de 2011

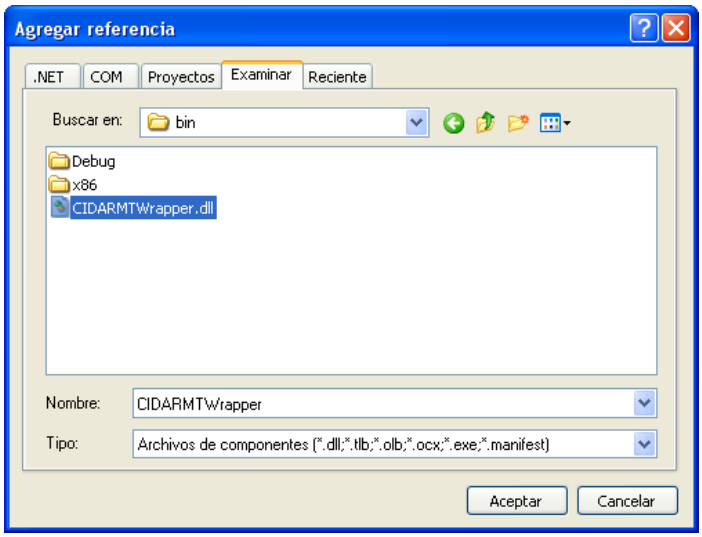
Tabla de contenido

Introducción

El presente documento es el manual de referencia del ensamblado “CIDARMTWrapper” que encapsula de una forma sencilla para su uso desde .NET las funcionalidades de la librería CIDARMT (Container ID Automatic Reader Multi Thread) capaz de ejecutar hasta 8 procesos simultáneos de reconocimiento de códigos de contenedor CIDAR. Para evitar problemas de concurrencia todos los métodos son estáticos, garantizando el acceso simultáneo a la clase, como por ejemplo añadir imágenes o leer los resultados en paralelo.

Para poder utilizar el ensamblado, los pasos a seguir son los siguientes:

1. Crear o abrir proyecto existente en Visual Studio
2. Agregar referencia
3. 



4. Declarar su uso en el código, mediante la instrucción adecuada para cada lenguaje:

   “Imports CIDARMTWrapper” (vb.net)

   “using CIDARMTWrapper” (C#)

   Etc.

5. A partir de este momento podemos utilizar sus funciones, que detallamos a continuación.

Declaración / Configuración

Configuración

El ensamblado .NET está preparado para funcionar como componente estático, facilitando así la concurrencia al mismo, incluso desde diferentes hilos de programación. Por ello no es necesario declarar ninguna variable. Este es un cambio importante en la versión 2.0 del ensamblado, respecto la versión 1.0 del mismo, que era parcialmente estático.

Las funciones de lectura preparan el motor de reconocimiento con la configuración pasada por parámetro y generan las peticiones de lectura, encolándose para ser procesadas por el gestor multitarea.

Así pues, el primer parámetro de todas ellas es una estructura inicializada con la configuración definida para esa petición. Esta inicialización puede realizarse a través del constructor o mediante una llamada posterior. En cualquier caso es obligatorio hacerla.

```
SetConfiguration ( ByVal Miliseconds As Int32,
                   ByVal aplicarCorreccion As Int32,
                   ByVal distance As Single,
                   ByVal vertCoef As Single,
                   ByVal horizCoef As Single,
                   ByVal radialCoef As Single,
                   ByVal angle As Single,
                   ByVal left As Int32,
                   ByVal top As Int32,
                   ByVal width As Int32,
                   ByVal height As Int32,
                   ByVal numSteps As Int32
                   ByVal steps() As Integer,
                   ByVal enableExtraInfo As Int32
                   )
```

Parámetros

Miliseconds	Indica al motor de reconocimiento un tiempo de proceso máximo tras el cual retornará con el resultado que haya obtenido hasta el momento. Este valor está expresado en milisegundos. Se recomienda 0
AplicarCorreccion	Corrección Coeficientes: En caso de poner este booleano a true, los 4 parámetros siguientes se aplicarán. Para más información sobre estos parámetros, dirigirse al manual de usuario de CIDAR.

distance	Parámetro para la distancia.
vertCoef	Parámetro para el coeficiente vertical.
horizCoef	Parámetro para el coeficiente horizontal.
radialCoef	Parámetro para el coeficiente radial (Efecto Ojo de pez).
angle	Parámetro para el ángulo.
left	Si alguno de los 4 parámetros (left, top, width, height) es diferente de cero, se utilizará esta ventana en vez de toda la imagen para buscar el código de contenedor. Left indica la coordenada izquierda de la ventana.
top	Coordenada superior de la ventana.
width	Anchura de la ventana.
height	Altura de la ventana.
numSteps	En caso de poner este valor >0 la se usaran las alturas del vector siguiente.
steps()	Array de alturas.
enableExtraInfo	Booleano indicando si queremos buscar la información extra del contendor.

Ejemplo 1

```
              ' VB.net code
Private Sub SetCIDARMTConfiguration()
        Dim stepsArray(2) As Int32
        If Me.CheckBoxCharacterHeight.Checked Then
            'Use Automatic height
            stepsArray(0) = 25
            stepsArray(1) = 60
            lCharSize = -1
        Else
            'Use user height
            stepsArray(0) = Convert.ToInt32(Me.userHeigth.Split(",")(0).Trim)
            stepsArray(1) = Convert.ToInt32(Me.userHeigth.Split(",")(1).Trim)
        End If
       CidarMTInterface.SetConfiguration(0, 0, 0, 0, 0, 0, 0,
              Convert.ToInt32(Me.NLeft.Value),
              Convert.ToInt32(Me.Ntop.Value),
              Convert.ToInt32(Me.NWidth.Value),
              Convert.ToInt32(Me.NHeight.Value), 2, stepsArray)
End Sub
```

Ejemplo 2

```
// C# code
Int32[] stepsArray = { 35, 65 };

CidarMTInterface.SetConfiguration(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 2, stepsArray);
```

Inicialización / Finalización

Init

Inicializa el **Container ID Automatic Reader** (**CIDAR**). Carga las Redes Neuronales Artificiales del OCR e inicializa parámetros. Esta función debe llamarse antes de usar cualquiera de las otras funciones de esta librería excepto SetConfiguration, que puede llamarse en cualquier momento.

long **Init** (long *lAverageCharacterHeight,*

long *bDuplicateLines* = false*,*

bool *bTrace* = false.

bool *wTrace* = false);

Parámetros

lAvCharacterHeight	Altura media aproximada de los caracteres de las códigos de contenedor a leer (en *píxeles*). Si pasamos un **-1** la librería calcula de manera automática la altura usando un rango predefinido de alturas. Si se selecciona altura automática, el tiempo de proceso aumenta considerablemente.
bDuplicateLines	Para imágenes capturadas con la mitad de las líneas pasaremos el valor 1 (true). Para los demás casos 0 (false).
bTrace	Traza interna. Este parámetro debe valer false.
wTrace	Traza interna. Este parámetro debe valer false.

Valor de Retorno

0 → Error.

1 → Ok.

long **Init** ();

Realiza la llamada anterior, con los valores por defecto

Init (-1, 0, 0, false)

Ejemplo

```
// C# code
CidarMTInterface.SetConfiguration (0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 2, stepsArray);
duplicateLines = 0;
return CidarMTInterface.Init(lCharSize, duplicateLines, 0, false);
```

QueryEnd

Libera la memoria reservada para el **Container ID Automatic Reader Multi Thread (CIDARMT)**.

Para ello espera a que se vacíe la cola de peticiones de reconocimiento acumulada y después libera los recursos. Esta función puede tardar un tiempo significativo en finalizar si se encuentran muchas peticiones pendientes de procesar.

Ejemplo

```
// C# code
// Stop CIDAR when finished processing images

if (CidarMTInterface.IsRunning())
{
      while (CidarMTInterface.IsProcessing)
      {
            Application.DoEvents();
      }
      CidarMTInterface.QueryEnd();
}
```

Solicitud de lectura de códigos de contenedor

Add

Añade el código de contenedor a la cola de peticiones de lectura. Cuando le llegue su turno se intentará leer el código de contenedor contenida en una imagen. Esta función recibe como entrada una imagen o la ruta en la que se encuentra una imagen. **Analiza la imagen** en busca de un código de contenedor y si la encuentra, **lee el código de contenedor** y colocará el resultado en una cola de códigos de contenedor resultados accesibles. Estas funciones retornan ID que servirá de identificador de la petición que se está procesando para la recuperación de resultados. Esta función puede llamarse de dos formas diferentes.

long **Add** (Bitmap *image*)

long **Add** (Bitmap *image,* Object *obj*)

long **Add** (Bitmap *image,* Bool *saveBitmapInfo* , String *image,* Object *obj*)

long **Add** (String *image*)

long **Add** (String *image,* Object *obj*)

long **Add** (Long *width*, Long *height*, IntPtr *image*)

long **Add** (Long *width*, Long *height*, IntPtr *image,* Object *obj*)

Parámetros

image — Este parámetro contiene la imagen a procesar en formato Windows Bitmap o la ruta en la que se encuentra la imagen o puntero a posición de memoria de la imagen.

saveBitmapInfo — Parámetro booleano (true / false) que indica si queremos almacenar en los resultados la imagen origen de la que se obtiene el código de contenedor. Si se activa a true, la imagen se podrá obtener con el método *GetSourceImage*() de la clase CodeInfo.

obj — Este parámetro contiene la referencia a un objeto cualquiera que queremos pasar a través de todo el proceso de reconocimiento, podemos recoger de nuevo este objeto al recuperar el resultado de la lectura. Este objeto puede

contener por ejemplo, información GPS del lugar de la captura, cámara que realizó la captura, etc.

Valor de Retorno

- **-1** → Error o cola de procesamiento llena.
- Número positivo, incluido cero → Ok.
    - El número indica el identificador asociado a la imagen, para su posterior uso.

Ejemplo

```
// C# code
Bitmap bitmap = eventArgs.Frame;

if (CidarMTInterface.IsRunning())
{
     CidarMTInterface.Add(bitmap,true,"");
}
```

Consultas disponibles

GetQueueSize

Nos indica el número de elementos en cola que tenemos pendientes de procesar por el motor de reconocimiento de códigos de contenedor. Imágenes añadidas (Add) que aún no han sido procesadas.

long **GetQueueSize**();

Valor de Retorno

- Número de elementos pendientes de procesar por el motor de reconocimiento.

Ejemplo

```
// C# code
// Wait to end processing images
while (CidarMTInterface.GetQueueSize > 0)
        {
            Application.DoEvents();
        }
```

IsRunning

Nos indica si el motor de reconocimiento de códigos de contenedor ha sido inicializado correctamente y no se ha detenido mediante el método QueryEnd.

bool **IsRunning**();

Valor de Retorno

- Cierto o falso dependiendo si el motor está listo para recibir peticiones de lectura.

Ejemplo

```
// C# code
// Add images while running

while (CidarMTInterface.IsRunning)
        {
            CidarMTInterface.Add(image);
        }
```

IsProcessing

Nos indica si el motor de reconocimiento de códigos de contenedor está pendiente de procesar resultados. Sea porque tiene elementos en cola o en proceso actualmente.

bool **IsProcessing**();

Valor de Retorno

- Cierto o falso dependiendo si el motor está procesando peticiones de lectura.

Ejemplo

```
// C# code
// Stop when ends processing


if (CidarMTInterface.IsRunning())
{
      while (CidarMTInterface.IsProcessing)
      {
            Application.DoEvents();
      }
      CidarMTInterface.QueryEnd();
}
```

GetLicensedCores

Nos indica el número núcleos de procesador disponibles para los que tenemos licencia válida. Será el número máximo de procesos simultáneos que podemos gestionar.

long **GetLicensedCores**();

Valor de Retorno

- Número de núcleos de procesador que tenemos licenciados para usar.

Ejemplo

```
// C# code
// Busy cores
labelCU.Text = String.Format("Cores used: {0}",
CidarMTInterface.GetLicensedCores - CidarMTInterface.GetFreeCores);
```

GetFreeCores

Nos indica el número núcleos de procesador disponibles. Será el número máximo de procesos simultáneos que podemos gestionar.

long **GetFreeCores**();

Valor de Retorno

- Número de núcleos de procesador que tenemos libres en el momento de la consulta.

Ejemplo

```
// C# code
// Busy cores
labelCU.Text = String.Format("Cores used: {0}",
CidarMTInterface.GetLicensedCores - CidarMTInterface.GetFreeCores);
```

Obtención de resultados

GetNumberOfCodes

Nos indica el número de códigos de contenedor que tenemos en la cola de resultados. Si el valor es mayor a cero, nos indica que tenemos algún código de contenedor, cuyo resultado aún no hemos leído.

long **GetNumberOfCodes**();

Valor de Retorno

Número de códigos en la cola de resultados.

Ejemplo 1

```
' vb.NET code
Dim ptinfo As CIDARMTWrapper.CodeInfo
While CidarMTInterface.GetNumberOfCodes > 0

      ptinfo = CidarMTInterface.GetFirstCode()

      Dim codeNumber As [String] = ptinfo.GetCodeNumber()

End While
```

Ejemplo 2

```
// C# code


CIDARMTWrapper.CodeInfo ptinfo = default(CIDARMTWrapper.CodeInfo);

while (CidarMTInterface.GetNumberOfCodes() > 0)
{
      ptinfo = CidarMTInterface.GetFirstCode();

}
```

GetFirstCode

Retorna la información de la primera código de contenedor que se encuentra en la cola de resultados y elimina esta información de la cola de resultados pendientes, por lo que si no la almacenamos, esta información se perderá.

CodeInfo **GetFirstCode**();

Valor de Retorno

Información del primer código de contenedor en la cola de resultados. En caso de no haber ningún resultado pendiente, se retorna null (o Nothing).

Estructura CodeInfo

Los métodos disponibles en la clase CodeInfo son los siguientes

GetLeftCodePosition ( )	Esta propiedad retorna la coordenada izquierda del código de contenedor encontrada.
GetRightCodePosition ( )	Esta propiedad retorna la coordenada derecha del código de contenedor encontrada.
GetTopCodePosition ( )	Esta propiedad retorna la coordenada superior del código de contenedor encontrada.
GetBottomCodePosition ( )	Esta propiedad retorna la coordenada inferior del código de contenedor encontrada.
GetCodeNumber ( )	Esta propiedad retorna el texto con el código de contenedor encontrada.
GetSourcePath ( )	Esta propiedad retorna la ruta (si disponible) de la imagen usada para buscar la código encontrada.
GetSourceImage ( )	Esta propiedad retorna la imagen usada para reconocer la código si se encuentra disponible.
GetId ( )	Esta propiedad retorna el identificador asignado a esta solicitud de lectura de código de contenedor, es el valor retornado por el método Add.
GetGlobalConfidence ( )	Esta propiedad retorna un valor que expresa la fiabilidad de la lectura de el código de contenedor analizada por el Container ID Automatic Reader. Este valor está expresado en tanto por ciento (0% – 100%).
GetAverageCharacterHeigth ( )	Esta propiedad retorna el valor medio de la altura de los caracteres de el código de contenedor encontrada.

GetNumCharacters ( )	Esta propiedad retorna el número de caracteres de el código de contenedor encontrada.
GetCharConfidence ( )	Esta propiedad retorna un array conteniendo las fiabilidades de cada uno de los caracteres de el código de contenedor encontrada. El array va de cero hasta *GetNumCharacters() -1.*
GetCodePosition ( )	Esta propiedad retorna las coordenadas del recuadro en el que se encuentra el código.
IsCodeVerified ( )	Esta propiedad retorna un valor entero indicando si en la lectura del código se ha validado correctamente el dígito de control.
GetExtraInfoCodePosition ( )	Esta propiedad retorna las coordenadas del recuadro en el que se encuentra el código extra.
GetExtraInfoCodeNumber ( )	Esta propiedad retorna el texto con el código extra de contenedor encontrada.
IsExtraInfoFound ( )	Esta propiedad retorna un valor booleano indicando si en la lectura del código se he encontrado información extra.
GetExtraInfoNumCharacters ( )	Esta propiedad retorna el número de caracteres de el código extra de contenedor encontrado.
GetExtraInfoConfidence ( )	Esta propiedad retorna un valor que expresa la fiabilidad de la lectura de el código de contenedor extra analizada por el Container ID Automatic Reader. Este valor está expresado en tanto por ciento (0% – 100%).
GetExtraInfoCharConfidence ( )	Esta propiedad retorna un array conteniendo las fiabilidades de cada uno de los caracteres de la información adicional del código de contenedor encontrado. El array va de cero hasta *GetNumCharacters() -1.*
GetNumLines ( )	Esta propiedad retorna el número de líneas para los códigos horizontales, puede ser 1 ó 2 líneas.
IsVertical ( )	Esta propiedad retorna si el último código leído es vertical o no.
IsInverted ( )	Esta propiedad retorna si el último código leído es invertido o no (Letras claras sobre fondo oscuro).
Dispose ( )	Importante propiedad que libera los recursos utilizados por la clase.

Ejemplo

```
// C# code
CIDARMTWrapper.CodeInfo ptinfo = default(CIDARMTWrapper.CodeInfo);
while (CidarMTInterface.GetNumberOfCodes() > 0)
{
        ptinfo =CidarMTInterface.GetFirstCode();
        if (Convert.ToDecimal(ptinfo.GetGlobalConfidence) > 60)
        {
        String codeNumber = ptinfo.GetCodeNumber;
        String lastNumber = "";
        if (this.listBoxResults.SelectedItem != null)
                lastNumber = this.listBoxResults.SelectedItem.ToString();
        if (!String.Equals(codeNumber, lastNumber))
                {
                this.listBoxResults.Items.Add(ptinfo.GetCodeNumber);
                this.listBoxResults.SelectedIndex = this.listBoxResults.Items.Count - 1;
                }
        }
}
ptinfo = null;
```

Ejemplo 2

```
' vb.NET code

Dim ptinfo As CIDARMTWrapper.CodeInfo
WhileCidarMTInterface.GetNumberOfCodes > 0
        ptinfo =CidarMTInterface.GetFirstCode()
        Me.avgPtime = Me.avgPtime + ptinfo.GetProcessingTime
        If (ptinfo.GetGlobalConfidence > Me.NumericUpDownMinConf.Value) Then
                If Me.CheckBoxBeep.Checked Then
                System.Media.SystemSounds.Exclamation.Play()
                Me.fiabilidades(Me.activePanel).Text = String.Format("{0:00}%",
                ptinfo.GetGlobalConfidence)
                Me.fiabilidades(Me.activePanel).Visible = True
                Me.activePanel = (Me.activePanel + 1) Mod 4
                Me.LabelACH.Text = "Avg. Char Heigth: " +
                Convert.ToInt32(ptinfo.GetAverageCharacterHeigth).ToString
        End If
End While
```

GetCode

Retorna la información del código de contenedor que se encuentra en la cola de resultados con el identificador que se le pasa por parámetro y elimina esta información de la cola de resultados pendientes, por lo que si no la almacenamos, esta información se perderá.

CodeInfo **GetCode**(long id);

Parámetros

id — Este parámetro contiene el id de la imagen de la que queremos obtener el resultado. Puede haber más de un resultado para el mismo id (Varias códigos de contenedor en una misma imagen) se retorna el primer resultado encontrado.

Valor de Retorno

Información de el código de contenedor en la cola de resultados con el id (Retornado por la función Add) que le pasamos por parámetro. En caso de no haber ningún resultado con este id, se retorna null (o Nothing).

Ejemplo

```
' vb.NET code
If Not (cidar Is Nothing) Then
id = CidarMTInterface.Add(Me.BytesToImage(image))
        If (id >= 0) Then 'Success enqueued
                Do
                        System.Threading.Thread.Sleep(100)
                        code = CidarMTInterface.GetCode(id)
                Loop While code Is Nothing 'Loop until get result (First result)
                listOfCodes.Add(code.GetCodeNumber)

                Do
                        code =CidarMTInterface.GetCode(id)
                        If Not (code Is Nothing) Then listOfCodes.Add(code.GetCodeNumber)
                Loop While Not (code Is Nothing) 'Enqueue if more results available
        Else
                'listOfCodes.Add(String.Format("Process failed."))
        End If
Else
        'Error Cidar
End If
```

Resultados: Métodos avanzados

Los siguientes métodos y funciones están indicados cuando se requiere mayor funcionalidad y acceso a la información una vez realizado el proceso de reconocimiento. Estos métodos permiten recuperar la información completa de los parámetros usados y de cualquier otra información añadida en el momento de realizar la llamada a la función Add. Permite conocer la información de fecha y hora exactas en que se añadieron las imágenes a la cola de procesamiento y otra información, normalmente no necesaria pero importante cuando se realizan operaciones de detección avanzados.

Si se usan varias cámaras y se quiere recuperar la información de los parámetros usados o la información adicional, estos métodos proporcionan esta funcionalidad adicional.

Podemos incluir en la llamada de reconocimiento información como cámara usada, posición de GPS, etc. Esta información puede ser recuperada en el momento de consultar el resultado del reconocimiento.

Disponemos de funciones equivalentes para elementos de las que se disponen para códigos, como son *GetNumberOfElements*, *GetElement*, etc.

GetFirstElement

Retorna información completa de la primera imagen con todos los resultados disponible. Esto incluye los parámetros usados para el reconocimiento de la imagen, cualquier información adicional pasada mediante la función 'Add' e información de TODOS los códigos leídos en la imagen. Este método elimina esta información de la cola de resultados pendientes, por lo que si no se guarda se perderá.

NLInfo **GetFirstElement**();

Valor de retorno

Información de la primera imagen disponible en la cola de resultados. Si no hay ningún resultado pendiente, se retornará 'null' (o 'Nothing').

Estructura NLInfo

Los métodos disponibles en la clase NLInfo son los siguientes:

GetId ( )	Esta propiedad retorna el id de la imagen procesada.
Configuration ( )	Esta propiedad retorna la estructura de información MTConfiguration usada cuando se realizó la llamada al motor de reconocimiento. Ver la documentación 'Vparmt – Vpar multithread' para disponer de más información sobre esta estructura de datos.
GetFirstItem ( )	Esta propiedad retorna el primer resultado CodeInfo de la imagen.
ItemCount ( )	Esta propiedad retorna el número de códigos disponible en la imagen procesada.
ReadItems ( )	Esta propiedad retorna el número de códigos leídas de la imagen procesada.
ImageInformation ( )	Esta propiedad retorna la estructura de información PicInformation de la imagen procesada.
Dispose ( )	Importante propiedad que libera los recursos utilizados por la clase. También libera los recursos de sus hijos, como la clase PicInformation.

Estructura PicInformation

Los métodos disponibles en la clase PicInformation son los siguientes:

GetId ( )	Esta propiedad retorna el id de la imagen procesada
GetPath ( )	Esta propiedad retorna la ruta o 'Nothing' (Si no hay ruta disponible) de la imagen procesada.
GetImage ( )	Esta propiedad retorna el Bitmap o 'Nothing' (si el bitmap no se encuentra disponible) de la imagen procesada. Los requisitos de memoria cuando se almacena información de Bitmap son MUY ELEVADOS por lo que se ha de usar con cuidado y bajo el propio riesgo.
GetObject ( )	Esta propiedad retorna el Objeto pasado como parámetro cuando se ha usado la función 'Add'.
GetTimeStamp ( )	Esta propiedad retorna la fecha y hora del momento en el que la imagen se añadió a la cola de procesamiento.
GetUTCTimeStamp ( )	Como en la propiedad anterior, pero se usan Fecha y hora universales.
Dispose ( )	Importante propiedad que libera los recursos utilizados por la clase.

Ejemplo

```
Dim ptElement As NLInfo = Nothing
'Get firs element where all results are ready
ptElement = MTInterface.GetFirstElement
While Not ptElement Is Nothing
      Dim ptinfo As CodeInfo = Nothing
      ptinfo = ptElement.GetFirstItem ()

      While Not ptinfo Is Nothing
            DisplayInfo(ptinfo)
            'Retrieve GPS information stored when Adding image
            Dim gpsCoordinates As GPS =
            ptElement.ImageInformation.GetObject
            ptinfo = ptElement.GetFirstItem ()
            Application.DoEvents()
      End While

End While
```