

Eym Barcode Reader ActiveX Control v2.4

EymBarcodeReader.ocx is an ActiveX control performs all necessary image processing and decoding stages to locate and read a variety of standard 1D barcodes from a bitmap. You may integrate it within your own application in a development environment such as Visual Basic, where you can freely decide that the source of these bitmaps should be (e.g image files or live video captures) and what you may use the decoded barcode data for. Software Licensing

- Software Licensing
- EymBarcodeReader Control Reference
- Functionality and supported standards

Major changes from version 1.0:

The most notable functional enhancement consists of the addition of a “Thorough” mode, allowing the OCX to pull multiple barcodes from the same image.

This additional functionality required a slight modification to the OCX interface. In version 2.x, most output properties (data, confidence levels, location, etc) take an index value as argument.

Version 2.x also benefits from other improvements that increased the OCX’s accuracy and speed.

Version 2.1 has also added support for Codabar (a.k.a. Ames Code/USD-4/NW-7/2 of 7 Code) and Patch codes

Version 2.2 has been further optimized leading to drastic speed increase that are most noticeable in thorough mode when processing large image files.

Version 2.3 has added support to color maps so that the host computer no longer needs to be set to true color (24 or 32 bits). This revision also added a new stage of “reality check” which should both decrease the occurrences of false positives and also slightly speed up the processing of large image files.

Version 2.4 has added a few controls that further optimize the system’s speed and accuracy. It also fixed an intermittent bug in “Fast mode”.

Software Licensing

This ActiveX control is provided on a “one time fee” basis. A proper design license is required in order to use the control within your own Visual Basic project and compile them. Once your project is compiled however, you are granted a limited right to package and distribute the OCX’s runtime as an integral part of your software without requiring that you purchase any further license.

Licensing Agreement

EymBarcodeReader Development Toolkits

GRANT OF LICENSE. Through this Agreement (“LICENSE”), Eric Metois (“EYM”) permits you (“LICENSEE”) the right to use one copy of EymBarcodeReader for development purposes on any single computer, provided EymBarcodeReader is in use on only one computer at any time. EymBarcodeReader is “in-use” on a computer when it is loaded into temporary memory (i.e. RAM) or installed into the permanent memory (i.e. hard disk, CD-ROM, or other storage device) of that computer.

REDISTRIBUTION OF EYMBARCODEREADER. To obtain rights to distribute EymBarcodeReader as part of the LICENSEE’s software application or derivative works (“PRODUCT”), you must acknowledge your agreement with, and compliance to, the terms of this license. Upon your acceptance of the terms of this License Agreement, EYM grants the LICENSEE, a limited, non-exclusive, right to use, reproduce, display or otherwise distribute or transfer limited copies of EymBarcodeReader runtimes, as an integral part of PRODUCT, in executable form only provided that:

- a) If the use is for corporate in-house development project(s), purchase of this development kit grants you a single site license with unlimited runtime distribution, within your derivative work(s), within a single site (physical location or address which does not exceed more than one square kilometer).
- b) The LICENSEE’s product is targeted to end users.
- c) The LICENSEE’s PRODUCT is not a development tool. PRODUCTs that can be used in other products (such as ActiveX controls) are considered to be development tools.
- d) You do not use EYM's name, logo, or trademarks to market the LICENSEE’s software application without prior written approval of EYM.
- e) You agree to indemnify, hold harmless, and defend EYM from and against any claims or lawsuits, including attorney's fees, that arise or result from the use or distribution of your software application.
- f) You provide EYM with a list of names of all commercially available software (current and future) that includes the EymBarcodeReader runtime.
- g) Only the EymBarcodeReader runtime can be distributed. Distribution of any EYM documentation, source code, or the distribution unlock codes, is a violation of this agreement.
- h) LICENSEE includes a statement substantially similar to the following within derivative work’s documentation and about box: "This product contains portions of image processing code owned and copyrighted by Eric Metois, Ph.D., (www.metois.com)."

If ANY of the above terms are not applicable to LICENSEE’s situation, or any of the above cannot be complied with, or you feel that you need modifications to this license, for any reason; then you must obtain an expanded direct license from EYM for your derivative works.

EYM reserves the right to charge the LICENSEE ten dollars (\$10.00) per unlicensed copy of PRODUCT. The license granted herein and all of your rights to use or maintain possession of EymBarcodeReader shall terminate immediately upon breach of any material provisions of this Agreement.

This license grants rights to LICENSEE for only the EYM code (EymBarcodeReader) mentioned above and does not convey any other rights of use or distribution to EYM technology.

COPYRIGHT. The Software is owned by EYM and is protected by United States copyright laws and international treaty provisions. LICENSEE may not remove or alter the copyright notice from any copy of EymBarcodeReader or any copy of the written materials, accompanying EymBarcodeReader.

OTHER RESTRICTIONS. This EYM License Agreement is your proof of license to exercise the rights granted herein and must be retained by you. LICENSEE may not rent or lease EymBarcodeReader. LICENSEE may not reverse engineer, decompile or disassemble EymBarcodeReader except to the extent such foregoing restriction is expressly prohibited by applicable law.

NO WARRANTY. ANY USE BY LICENSEE OF EYMBARCODEREADER IS AT THE LICENSEE'S OWN RISK. EYMBARCODEREADER IS PROVIDED FOR USE ONLY WITH WINDOWS PRODUCTS AND RELATED APPLICATION SOFTWARE AND/OR HARDWARE PRODUCTS. EYMBARCODEREADER IS PROVIDED FOR USE "AS IS" WITHOUT WARRANTY OF ANY KIND. TO THE MAXIMUM EXTENT PERMITTED BY LAW, EYM DISCLAIM ALL WARRANTIES OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, WITHOUT LIMITATION, IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. EYM IS NOT OBLIGATED TO PROVIDE ANY UPDATES TO EYMBARCODEREADER.

NO LIABILITY FOR CONSEQUENTIAL DAMAGES. In no event shall EYM be liable for any damages whatsoever (including, without limitation, incidental, direct, indirect and consequential damages, damages for loss of business profits, business interruption, loss of business information, or other pecuniary loss) arising out of the use or inability to use this EYM product, even if EYM has been advised of the possibility of such damages. Because some states/countries do not allow the exclusion or limitation of liability for consequential or incidental damages, the above limitation may not apply to you.

INDEMNIFICATION BY LICENSEE. If LICENSEE distributes EymBarcodeReader in violation of this Agreement, you agree to indemnify, hold harmless and defend EYM and its suppliers from and against any claims or lawsuits, including attorney's fees that arise or result from the use or distribution of EymBarcodeReader in violation of this Agreement.

Limited Warranty

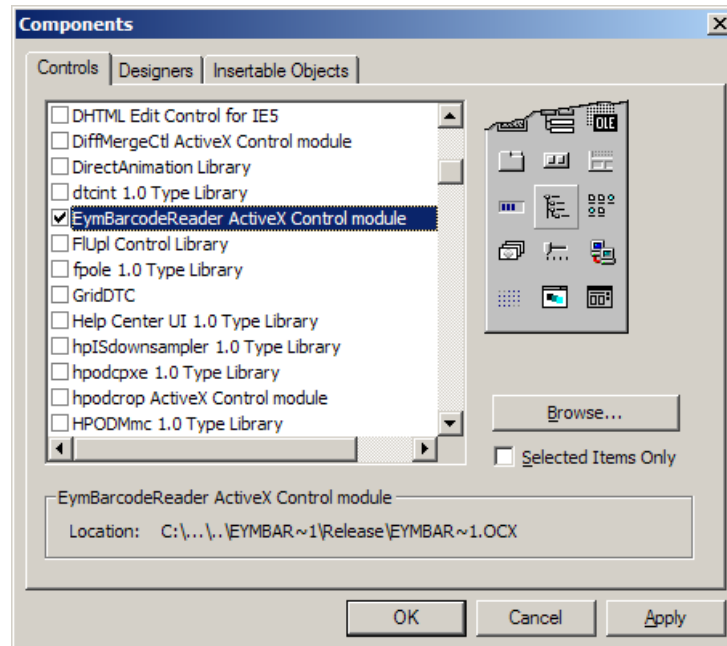
LIMITED WARRANTY. EYM warrants that the SOFTWARE will perform substantially in accordance with the accompanying written materials for a period of ninety (90) days from the date of receipt. Some states/jurisdictions do not allow limitations on duration of an implied warranty, so the above limitation may not apply to you.

CUSTOMER REMEDIES. EYM's entire liability and your exclusive remedy shall be, at EYM's option, either (a) return of the price paid, or (b) replacement of the SOFTWARE that does not meet EYM's Limited Warranty and which is returned to EYM with a copy of your receipt. This Limited Warranty is void if failure of the SOFTWARE has resulted from accident, abuse, or misapplication. Any replacement SOFTWARE will be warranted for the remainder of the original warranty period or thirty (30) days, whichever is longer.

NO LIABILITY FOR CONSEQUENTIAL DAMAGES. To the maximum extent permissible by applicable law, in no event shall EYM be liable for any damages whatsoever (including without limitation, damages for loss of business profits, business interruption, loss of business information, or any other pecuniary loss) arising out of the use of or inability to use this EYM product, even if EYM has been advised of the possibility of such damages. Because some states/jurisdictions do not allow the exclusion or limitation of liability for consequential or incidental damages, the above limitation may not apply to you.

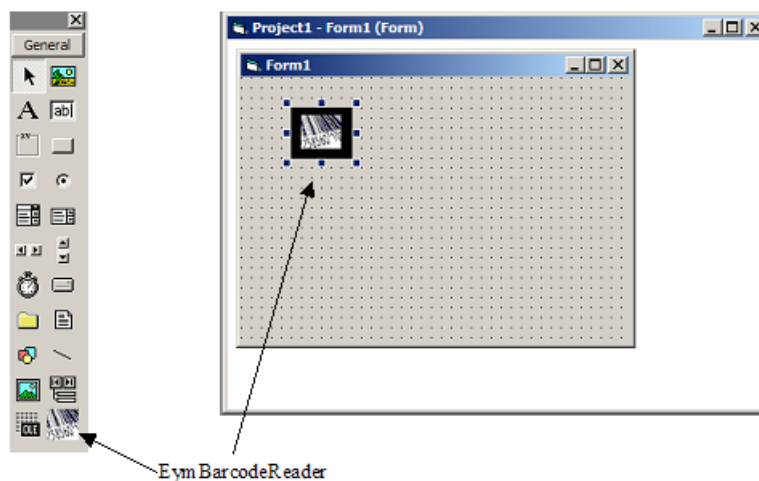
EymBarcodeReader Control Reference

Before you may use the EymBarcodeReader ActiveX control in your VB project, you must add it to your project using the Components dialog available from the Project menu.



EymBarcodeReader control in the Components dialog

The control will subsequently appear in your project's toolbox window. For there, you'll be able to add it to a form like any other insertable control.



EymBarcodeReader control in the toolbox window and an instance dropped onto a frame

Much like other common controls that don't have any intrinsic graphical element to them (such as timers for instance), this control will be invisible at runtime.

The control was designed with simplicity of use in mind. Most of this ActiveX control's properties are read-only as they are intended to convey information about the barcodes that were read. The control's main "input" is its Picture property, which is the means by which your application passes

an image to the control. It also counts four other Read/Write properties that are intended to configure the control's behavior.

Input properties

<i>“Input” Properties</i>	<i>Access</i>	<i>Description</i>
Picture	Read/Write	Property used to pass image data to the control.
ProtocolMask	Read/Write	Long. Binary mask describing the list of barcode protocols the control should be looking for.
ConfidenceThreshold	Read/Write	Single. Minimum confidence level (between 0.0 and 1.0) required in order to report a successful read.
ContrastSensitivity	Read/Write	Single. Minimum high spatial frequency energy level (between 0.0 and 1.0) required in order to attempt the decoding of a barcode.
ThoroughMode	Read/Write	Boolean. This Boolean value toggles the OCX between its “Fast” and “Thorough” modes.
maxNbCodes	Read/Write	Long. This optional parameters tells the system how many barcodes to expect within the image for optimization purposes.
ExpectedDpi	Read/Write	Long. This parameter is used to tell the system what image resolution to expect in dpi.
SearchGranularity	Read/Write	Long. Control how coarse the first barcode location analysis should be. The large this number of pixel and the coarser (and faster) the search.

The **Picture** property is a convenient way to pass image data to the core algorithms implemented in the EymBarcodeReader control from a your VB application. For example, your project may have a PictureBox that gets loaded with a still image (either from a file or another source) and passing that image to the barcode reader is done via a simple assignment:
EymBCrd.Picture = Picture1.Image

The **ProtocolMask** property provides a means to enable of disable the various barcode protocols supported by the EymBarcodeReader control. These various protocols enablers are encoded as follows:

<i>Value</i>	<i>Protocol</i>
1	EAN-13 and UPC-A + 2/5 digits extensions
2	CODE128 - includes code types UCC/EAN-128
4	CODE-39
8	CODE-93
16	EAN-8
32	UPC-E
64	UPC extensions (2 or 5 digits supplements)
128	Interleaved 2 of 5
256	CODABAR (aka Ames Code/USD-4/NW-7/2 of 7)
512	Patch Codes (1, 2, 3, 4, 6 and T)

These values are then combined to provide a binary mask describing any subset of enabled protocols. For example, in order to enable EAN13, EAN8 and UPC-E and disable all other protocols, we would set the ProtocolMask property to:
`EymBCrd.ProtocolMask = 1 + 16 + 32`

By default and when a new instance of the control is created, this mask is set to 191, enabling all of the protocols except UPC extensions when they stand by themselves.

The value of the **ConfidenceThreshold** property tells the control how confident it should be about its decoding results before deeming them as successful. This value lives between 0.0 (very loose) and 1.0 (virtually unattainable). By default and when a new instance of the control is created, this threshold value is set to 0.81 (i.e. 81% confidence required in order to report any decoding result).

The value of the **ContrastSensitivity** property tells the control to only pay attention to areas of the image that score above that threshold value in terms of contrast and high spatial frequency energy. This parameter is a number between 0.0 and 1.0. The default value for this threshold is 0.07 (i.e. 7% of the input image's color range).

The value of the **ThoroughMode** property tells the control whether or not it should perform a more thorough search for potentially multiple barcodes within the images we send to it. If the property is set to false, the control will operate in its "Fast" mode wherein it will look for the single most likely location where a barcode might be and attempt to read data from that location only.

Note about Thorough Mode and Patch Codes: Even when it is set to operate in its Thorough Mode, the control will never report more than one PatchCode per image.

The **maxNbCodes** property can be used to speed up the system in Thorough Mode. In version 2.4 this mode was further optimized and it you may further speed up the process if you know in advance how many barcodes are contained within the image.

The **ExpectedDpi** property is used to tune the system to the image resolution you use in your application. Originally the system was targeting image resolutions in the 100 to 200 dpi. As it turns out many users are using this control to process images that were scanned at much higher resolution. The system doesn't need to know exactly what the image resolution is but you can help it do a better and faster job by providing a fair guess in that field.

The **SearchGranularity** parameter is primarily used to speed up the system. Although EymBarcode doesn't do a traditional "scan-line" process, there is still a concept of search granularity when it first analyses the provided image in order to assess where potential barcodes might live. If you deal with large and high-resolution images, you can get away with a coarser granularity (i.e. larger number), which will subsequently speed up the process.

Output properties

<i>"Output" Properties</i>	<i>Argument</i>	<i>Access</i>	<i>Description</i>
NbCodesFound	none	Read only	Integer. Number of barcodes that were successfully decoded from the provided image. All following properties will require an index value as an argument. The index associated with the first code is 1. The index of the last code is NbCodeFound .
RawData(index)	Short	Read only	String. Raw data payload conveyed by the decoded barcode (referenced by index).
Symbology(index)	Short	Read only	String. Name of the decoded barcode's

symbology (referenced by **index**).

BarcodeType(index)	Short	Read only	String . Name of the decoded barcode's code type (referenced by index).
DataDescriptor(index)	Short	Read only	String . Description of the data conveyed by the decoded barcode (referenced by index).
Data(index)	Short	Read only	String . Formatted payload conveyed by the decoded barcode (referenced by index).
Confidence(index)	Short	Read only	Single . Confidence measure (between 0.0 and 1.0) associated with the decoded data (referenced by index).
Cx(index)	Short	Read only	Long . Distance (in pixels) between the left edge of the image and the center of the barcode (referenced by index).
Cy(index)	Short	Read only	Long . Distance (in pixels) between the top edge of the image and the center of the barcode (referenced by index).
StartX(index)	Short	Read only	Long . Distance (in pixels) between the left edge of the image and the starting edge of the barcode (referenced by index).
StartY(index)	Short	Read only	Long . Distance (in pixels) between the top edge of the image and the starting edge of the barcode (referenced by index).
StopX(index)	Short	Read only	Long . Distance (in pixels) between the left edge of the image and the stop edge of the barcode (referenced by index).
StopY(index)	Short	Read only	Long . Distance (in pixels) between the top edge of the image and the stop edge of the barcode (referenced by index).

The **RawData**, **Symbology**, **BarcodeType**, **DataDescriptor** and **Data** properties are human readable strings that are set to the empty string ("") if the provided index is smaller than 1 or greater than **NbCodesFound**.

In order to assess whether or not any valid codes were found we must first examine the object's **NbCodesFound** property.

If a valid code was found then the object's **Cx**, **Cy**, **StartX**, **StartY**, **StopX** and **StopY** properties provide the location of the decoded barcode within the provided image. These distance measurements are provided in number of pixels.

Example

This simple VB6 example assumes that you have a picture box named "Picture1", a label named "lblResult" and a EymBarcodeReader named "EymBCrd" on your frame.

We set the Picture's Autosize property to True so that it will automatically resize itself when we load a image from a file. We also set it's scale mode to 3 (i.e. pixel mode) as we intend to draw on it using pixels as our distance measures.

```
(...)  
Picture1.AutoSize = True  
Picture1.ScaleMode = 3 ' i.e. pixels  
(...)
```


We could leave the barcode reader object's configuration in its default stage but as an illustration, we will force the confidence threshold to 0.75 (i.e. 75%) and restrict the search protocols to EAN13, EAN8 and UPC-E.

```
(...)  
EymBCrd.ConfidenceThreshold = 0.75  
EymBCrd.ProtocolMask = 1 + 16 + 32  
(...)
```

We further configure the OCX into its "fast" mode using its **ThoroughMode** Boolean property.

```
(...)  
EymBCrd.ThoroughMode = False  
(...)
```

We subsequently load an image file to our picture box, pass that image data to the barcode reader object and look for a valid code. If a valid code is found, we write the formatted data, the barcode type and the confidence level on the lblResult label, and further overlay some location feedback on the picture box.

```
(...)  
Picture1.Picture = LoadPicture("SomeBarcodeImage.jpg")  
EymBCrd.Picture = Picture1.Image  
If EymBCrd.NbCodesFound > 0 Then  
    lblResult.Caption = EymBCrd.Data(1) & " - " & EymBCrd.BarcodeType(1) & _  
        " - " & Format(100# * EymBCrd.Confidence(1), "00.0") & "%"  
    Picture1.Line (EymBCrd.startx(1), EymBCrd.starty(1))-(EymBCrd.stopx(1),  
EymBCrd.stopy(1)),vbRed  
    Picture1.Circle (EymBCrd.cx(1), EymBCrd.cy(1)), 10, vbBlue  
End If  
(...)
```

Important note for VB.net users concerning the Picture property

You can use the EymBarcodeReader OCX from within Visual Basic .Net using COM Interoperability. As VB.Net imports the OCX, it will wrap it up in a couple of interoperable DLLs. You should then be able to drop an instance of the OCX on a VB form like you would do in VB6. From then on, you should be able to use the object's various field the same way you would under VB6 with the exception of the Picture input property.

Indeed image handles have evolved from VB6 and VB.Net and in order to pass image data from VB.Net to the EymBarcodeReader OCX, you need to translate the .Net image handle back to its legacy format using the VB6.ImageToIPicture() function call:

```
(...)  
EymBCrd.Picture = VB6.ImageToIPicture(Picture1.Image)  
(...)
```

Functionality and supported standards

Image processing stage

Before attempting to decode any barcode from an image, the EymBarcodeReader control performs some image processing that aims to go beyond the potentially crude resolution of the provided image. This stage is key to the control's ability to read barcodes from images that are casually captured from a webcam. Still, the image should be reasonably focused on the object.

Supported barcode standards

Some barcode standards are actual symbologies (i.e. means to encode data graphically) while some others are encoding systems that live on top of symbologies. The EymBarcodeReader control is aware of the following variety of symbologies and encoding systems.

<i>Standard</i>	<i>Type</i>	<i>Description</i>
EAN-13	Symbology	Used with consumer products internationally, it conveys 13 decimal digits. The code consists of a 2 to 3 digit number system that identifies a numbering authority, a manufacturer and product code, and a single check digit. The EymBarcodeReader control is aware of 89 different numbering authorities, which it will return in human readable form in its DataDescriptor property.
UPC-A	EAN-13 subset	Used in consumer products in the US and Canada. These are EAN-13 codes with number systems start with "0". The EymBarcodeReader control is further aware of few UPC-A sub formats including UPC coupons.
JAN-13	EAN-13 subset	Used in consumer products in Japan.
ISSN	EAN-13 subset	International Standard Serial Number for periodical.
ISBN a.k.a. Bookland	EAN-13 subset	International Standard Book Numbering. The EymBarcodeReader control is further aware of the specific formatting of these numbers.
ISMN	EAN-13 subset	International Standard Music Number. The EymBarcodeReader control is further aware of the specific formatting of these numbers.
EAN-8	Symbology	Short version of the EAN-13 symbology, it conveys 8 decimal digits including a check digit.
UPC-E	Symbology	Short version of the UPC-A symbology, it conveys 8 digits including a check digit. Its data is a compressed from of a full UPC-A code. The EymBarcodeReader control will decompress that data and provide it as a standard UPC-A product identification code in its Data property.
Code 128	Symbology	High density and variable length symbology that includes a check digit. This symbology is capable of conveying a large set of characters and used extensively worldwide.
UCC/ EAN128	Encoding system	This encoding system lives on top of the Code 128 symbology. This standard is used to convey a wide range of common data including package dimensions, weights, expiration dates, coupons and more.

		The EymBarcodeReader control is aware of 91 different application identifiers, for which it will provide human readable descriptions in its DataDescriptor property and format its Data property appropriately.
Code 39	Symbology	Variable length alpha-numeric symbology with no enforced check digit.
Vehicle Identification Number	Encoding system	This encoding system lives on top of the Code 39 symbology. It consists of 17 alphanumeric characters that convey the manufacturer's identification (WMI), a descriptor of the vehicle's configuration (VDS), a check digit, the year it was built, the manufacturer's plant identification and the vehicle's serial number.
Code 93	Symbology	Higher density variant of Code 39. This code always includes a check digit.
Interleaved 2 of 5	Symbology	Variable length decimal symbology with no enforced check digit.
UPC 2-digit supplemental	Symbology	Conveys 2 additional decimal digits following an EAN-13, EAN-8, UPC-A or UPC-E barcode. This information is typically intended to identify the issue number of the product.
UPC 5-digit supplemental	Symbology	Conveys 5 additional decimal digits following an EAN-13, EAN-8, UPC-A or UPC-E barcode. This information is typically intended to convey pricing information. The EymBarcodeReader control is aware of 10 different currency codes.
Codabar	Symbology	Variable length symbology with no enforced check digit. It may encode 16 different characters, plus an additional 4 start/stop characters. This symbology is used by U.S. blood banks, photo labs, and on FedEx airbills.
Patch Codes	Symbology	Set of 6 distinct barcode patterns (1, 2, 3, 4, 6 and T) that are typically used as document separators.

Further details about these standards was graciously made available online at these sites:

VIS' **Barcode Island** - <http://www.barcodeisland.com/>

Russ Adam's **Barcode1** - <http://www.adams1.com/pub/russadam/barcode1.html>