How to Improve Barcode Reading Speed and Accuracy

The use of barcode technology has proven itself for decades and continues to see adoption growth. It's proven beneficial for:

- Conducting and managing data entry
- Asset, inventory, and information tracking
- Improving operational efficiency
- Reducing human and data errors

More and more applications and systems managers seek to use barcodes. However, once you head into choosing a barcode, you might be surprised to uncover how many barcode types exist on the market.

Also, while barcodes are widely used for their speed and efficiency, there are many potential technical obstacles and issues that can arise to create bottlenecks and inefficiencies. If you can cover critical basics when implementing barcodes technology, you'll set yourself on a desirable path to benefit from them.

I. The Basics of Barcode Reader Technology and Development

How Barcode Scanners Work

Understanding the technology behind a barcode scanner can be complex. So, let's simplify it.

The barcode reader software scans the entire width of a barcode image to identify a barcode. Once it scans a barcode, it decodes it and presents the encoded information. Also, considering the wide use of mobile devices nowadays, a camera is also used to capture the barcode image for processing and decoding its value.

There are mainly two types of barcodes:

1D barcodes consist of vertical black lines against an all-white background. The lines are of varying widths with specific gaps resulting in a particular pattern.

2D barcodes consist of square or rectangular patterns of two dimensions. Again, the patterns are usually black against an all-white background.

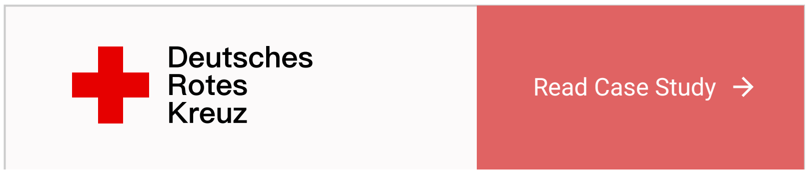
Generally, 2D barcodes store more data and support a bigger character set than 1D barcodes.

Where Barcodes are Used

Barcodes are used in countless applications across a myriad of scenarios from accounting/finance, government, healthcare, industrial, inventory, retail, transportation, and so on. Here are a few quick real-world examples.

ACCOUNTING

A large worldwide non-profit wanted a barcode scanning tool to use in accounting to increase productivity by automating certain tasks. To this end, they wanted to be able to start the tool, use it to read data off a barcode, and automatically rename associated PDF files the same as the barcode number. Initial research started with how to build the tool internally. The team quickly realized creating a barcode scanning application from scratch would be immensely time, resource and cost consuming. Today, the barcode addition is used to speed processing of invoices and related payments.

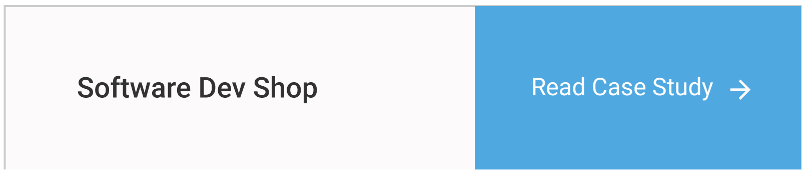


[German Red Cross Uses Dynamsoft Barcode Reader SDK to Quickly Automate Barcode-to-PDF Procedure]

HEALTHCARE

Barcodes are widely used in healthcare for dispensing medication and authenticating for security, specimen samples and more.

Barcodes can also be used to track devices given to a patient. This can work both ways. You can track a device to each patient it was given to and you can track who administered it. For the administrator, this can include a doctor, nurse or other staff. For tracking where the device came from, this can include suppliers or manufacturers. These trails in an EHR are all helpful in administering patient care or for managing device vendors and their transactions with you.



[A healthcare practice uses Dynamsoft's SDK for automation]

RETAIL

Some stores have evolved their barcode use by also building and integrating self-checkout apps. In these scenarios, customers can make purchases using their mobile phones while they are in a store. They can simply scan barcodes for product information and to make a purchase. Product information can be as rich as desired, to include information such as a description, unit price, place of origin, and more. This scan setup is commonly accomplished by using a barcode software development kit (SDK) to enable barcode scanning via smartphones. By allowing customers to self-add products to a virtual shopping cart while shopping in an aisle, retailers can eliminate making customers stand in line with their products at a physical checkout.

For retailers, this can also enable rich data capture for more targeted marketing. Customer shopping behaviors can be used to better optimize experiences. This can include product-based experiences, such as offering discounts for future purchases on previously purchased products. It can also include time-based marketing, such as targeting email newsletters or offers during a time a customer is more likely to visit a store.

Character Set

There is a lot to consider when attempting to select an appropriate barcode to use. You should familiarize yourself with the various barcode types, their capabilities, and requirements, to ensure a proper barcode selection for your given scenario.

To select the barcode for your given task, you need to understand its character set. There are generally three different types of character sets:

- **Numeric** — A numeric character set includes numbers only (0-9)
- **Alphanumeric** — An alphanumeric character set includes numbers and alphabetic characters (0-9 and A-Z)
- **Full ASCII** — An ASCII character set includes any ASCII character (value 0-127).

Barcode capabilities range anywhere from using dozens of characters to encode with a 1D barcode to thousands of characters in 2D barcodes. But bigger is not always better. Barcode selection should be based on efficiencies of maximizing a character set with a little room to grow rather than perhaps double or triple the character set you may use. Otherwise, this might have an impact on processing speeds.

II. Localization and Decode

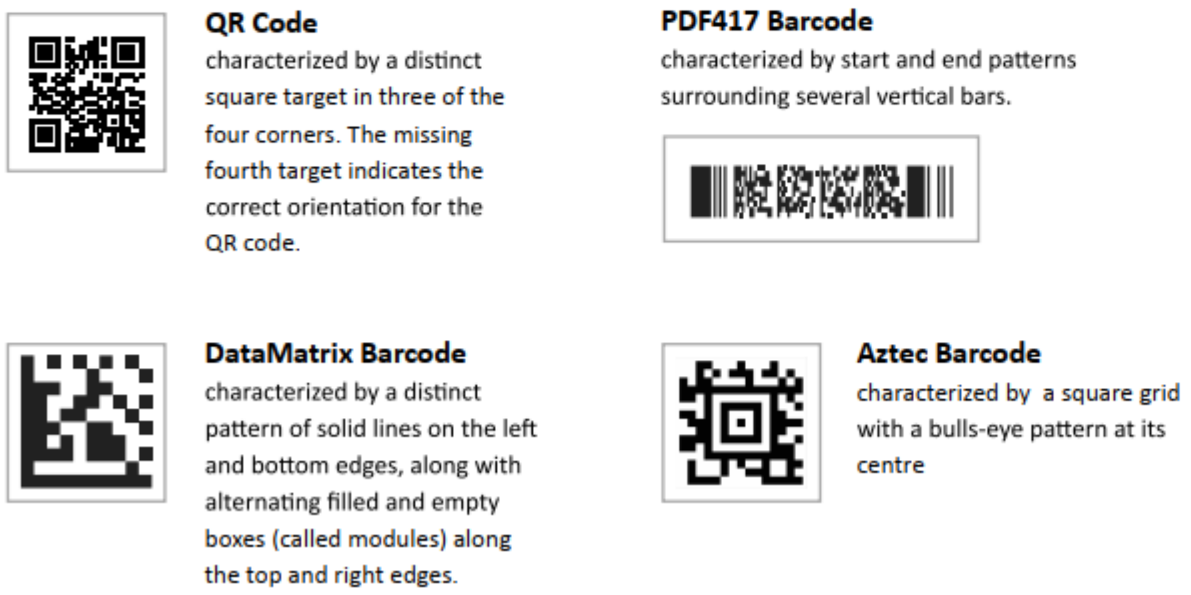
How to Decode 1D Barcodes

Upon initiating a scan, the scanner searches for any barcode to detect its type. A barcode reader scans the entire width of an image trying to identify whether there are barcode candidates — black/white patterns.

Once a barcode is identified, the barcode API has regions of interest to detect. Decoding is then initiated. In this step, the barcode scanner tries to understand the encoded information. It counts and compares image pixels to match start and end identifiers. Then, it interprets patterns between the start and end identifiers based on that code type's specifications to unravel the encoded data.

How to Decode 2D Barcodes

Similar to 1D barcodes, upon initiating a scan, the scanner tries to find a barcode. As mentioned, different barcode symbologies have different recognition patterns.



QR Code
characterized by a distinct square target in three of the four corners. The missing fourth target indicates the correct orientation for the QR code.

PDF417 Barcode
characterized by start and end patterns surrounding several vertical bars.

DataMatrix Barcode
characterized by a distinct pattern of solid lines on the left and bottom edges, along with alternating filled and empty boxes (called modules) along the top and right edges.

Aztec Barcode
characterized by a square grid with a bulls-eye pattern at its centre

In the decoding step, the barcode reader API tries to interpret patterns of the black/white modules based on that code type's specifications. These specifications are all standardized. Besides the barcode information, which is typically a combination of numeral and characters, a 2D barcode usually contains redundant data. The software is then able to read and decode the data value. The encoded value is built like stacks of codewords.

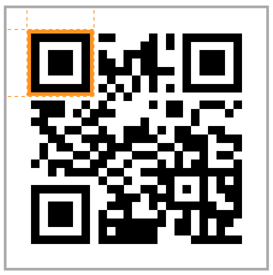
Advanced localization methods

There are 3 Localization methods:

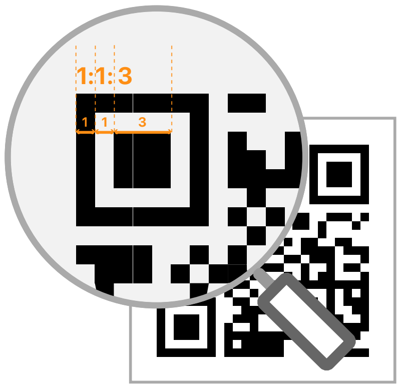
- Connected Blocks
- Statistics
- Line Segment

Each method has its unique advantages and disadvantages. A well-designed barcode scanner needs to balance the trade-off between speed and accuracy.

Connected Blocks
The first localization method is through identifying connected blocks in an image. It looks for a cluster of connected pixels . For example, each circled region is considered a block in the image below.

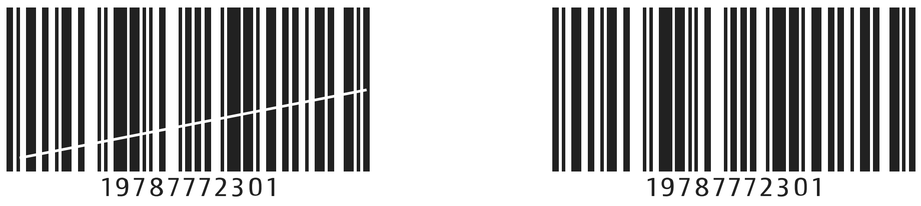


Once all the blocks in an image are identified, it then looks for known barcode patterns to filter the barcode locations. For instance, QR codes are characterized by a distinct square target in three of the four corners. Each square target in the corner consists of two blocks: a solid square in the middle of a hollow outer square. The cross-sectional thickness of the outer square, the gap and the width of the solid center square has the ratio of 1:1:3.



Once the 3 distinct squares are located, edges of the barcode need to be determined along with, grouping the 3 corner square targets to the same barcode, consider special cases where a square is missing or partially damaged etc. Once the barcode regions are identified based on barcode patterns, the rest of the space is filtered out.

The main advantage of this method is speed and low processing power required. Its drawback is the high susceptibility to damages on the barcode. For instance, if a pen marks across a 1D barcode horizontally, the separated bars are now one connected block and destroys the recognizable 1D pattern



Before After

Statistics
The main advantage of this localization method is identifying barcodes in blurred images. If an arbitrary region in a binarized image is selected to calculate the ratio between black and white pixels, we can conclude that a particular region is likely to contain a barcode or not. Once a region is concluded to contain a barcode it then filters the barcode regions for barcode location by searching for known barcode patterns.

Line Segment
As the name suggests this method looks for straight lines in an image and then filters out the barcode regions by matching the lines to known barcode patterns. This method works well for damaged barcodes. As a trade-off, it requires additional computation than the Connected Blocks method.

III. Developer Tip: Improve Barcode Recognition Speed

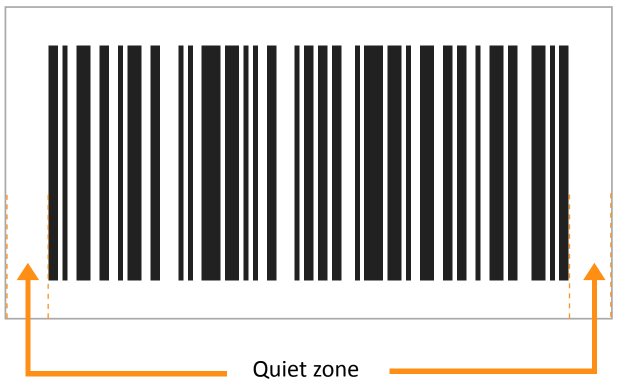
Location

The location of the barcode involves where it is physically located on a page, box or other items. It's important to consider user scenarios because location identification can increase recognition speed. Also, you want to consider location to ensure its placement isn't going to be where it's likely to become damaged. For example, if you're placing it on a printed health record, you probably want to avoid putting the barcode where users commonly staple another item with the barcoded item.

If you can specify a quadrant or area as to where the barcode will always be this can also improve speed. For example, if you know barcodes will always be in a three-inch by three-inch box at the top right corner of an 8.5" x 11" page, with good software you can instruct the software to read only that area. Thus, time isn't wasted scanning other parts of the page where barcodes are not present.

Quiet Zone

Part of having success in barcode scans is having a properly defined quiet zone. A quiet zone is a blank area or margin on either end of a barcode. That blank area tells a barcode scanner where the barcode starts and stops. Quiet zone specifications vary depending on the barcode symbol being used. Usually, at least an eighth of an inch is a minimal requirement. However, it's important to verify quiet zone requirements for any chosen barcode. Be sure the location where you'll place the barcode allows for required quiet zones.



Barcode Direction and Rotation

As you can imagine, specifying the direction your barcodes will be in when decoded can increase speed. This is because the software doesn't have to account for multiple orientations to look for. For example, if you can specify in your application that barcode orientations will always be horizontal, as opposed to vertical, speed will be nominally improved.

For further orientation-related speed improvements, you'll want to avoid as much as possible having to pre-process a barcode. While sometimes unavoidable, this means not having to de-skew it or perhaps smooth it.

Smooth-Zooming

Another related process that can impact decoding speed is when you must smooth a barcode. This essentially adds resolution and darkens damaged barcodes that are heavily faded and lacking pixel density. But this isn't just about duplicating pixels to darken it as often this can negatively impact barcode recognition.

5 46467 37757 1
5 46467 37757 1

Before Smooth-zoom

After Smooth-zoom

Smoothing a barcode that is lacking contrast or pixel density must be done intelligently and an appropriate zoom level should be employed in the barcode reader software. So, you can imagine that whenever smoothing is implemented, speed is sacrificed. Just as you want to avoid deskewing, you'll want to avoid smoothing too.

There are cases where fixing damaged barcodes are unavoidable parts of an operation for a specific application. In such cases, being able to pre-process barcodes will help speed operations. But, in most cases, avoiding pre-processing mechanisms is an ideal path to speedier decoding.

Barcodes per Page

Your barcode reader software should allow for parameters that define how many barcodes on a page will be scanned or how many pages need scanning per document. For example, let's say you have a five-page document with two barcodes and the barcodes are on page two and three. Being able to specify in your software how many barcodes are expected and on what pages can dramatically improve speed.

The speed benefits are compounded as the number of documents you have grows. So, you should be able to make the software scan only for a set number of barcodes or for particular pages. You can also set this on a per page-basis, such as if the barcodes scanned in a page has hit the maximum number of barcodes allowed.

Multi-Threading

The use of multi-threading by barcode reader software is also important to speed. Multithreading refers to running more than one thread of execution for a task (a program, or a process) within an operating system. Each request for the task is kept track of as a thread with a separate identity. The requests are processed in parallel in multi-cores or multiprocessors. Given that multi-core CPUs are so common now, it’s important for a software application to process information in an asynchronous fashion. In this way, a piece of software can use multithreading to speed things up without wasting your processor’s computing capability.

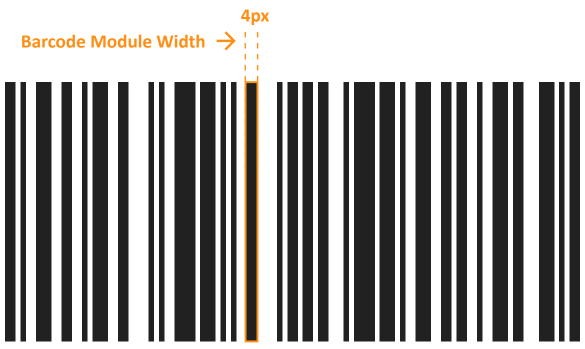
IV. Developer Tip: Improve Barcode Recognition Rates

Resolution

The quality of the barcode image produced by your chosen barcode generator matters. Generally, a barcode reader processes the counting of pixels in an area to determine the width and location of a particular bar in a bar code symbol. So, a lack of resolution can interfere with this process.

Usually, a minimum of 200 dpi is needed for acceptable barcode recognition. The higher you can go in resolution the better. Most organizations will seek to balance good barcode recognition with storage savings realized by minimizing resolution. Just be sure not to sacrifice barcode recognition efficiencies for storage efficiencies. Remember that the lower the resolution the more likely the symbol will lack the pixel density needed for good recognition.

Generally, for 1D barcodes, it's understood at least three pixels are needed per smallest bar and gap in a symbol. For 2D barcodes, it's usually around five pixels. Sometimes, despite proper resolution settings, good pixel density just isn't there. This could be for many reasons – poor quality labels or printer, and so on.

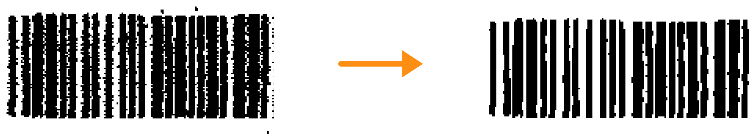


Code 128

Dealing with Unreadable or Damaged Barcodes

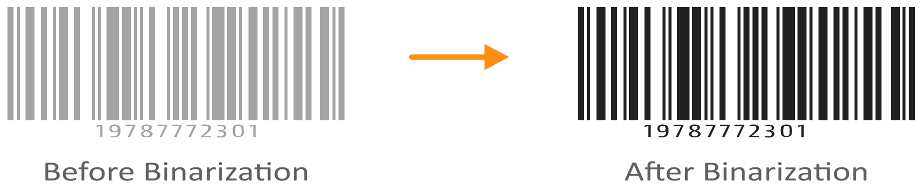
Barcode readers are only as good as the software behind them at scanning low-quality barcodes or damaged barcodes. There are three main problems to look for to determine if image enhancement of a barcode is needed via pre-processing:

Dilation — adds pixels to the boundaries of objects in an image, while erosion removes pixels on an object's boundaries.



Before Dilation   After Dilation

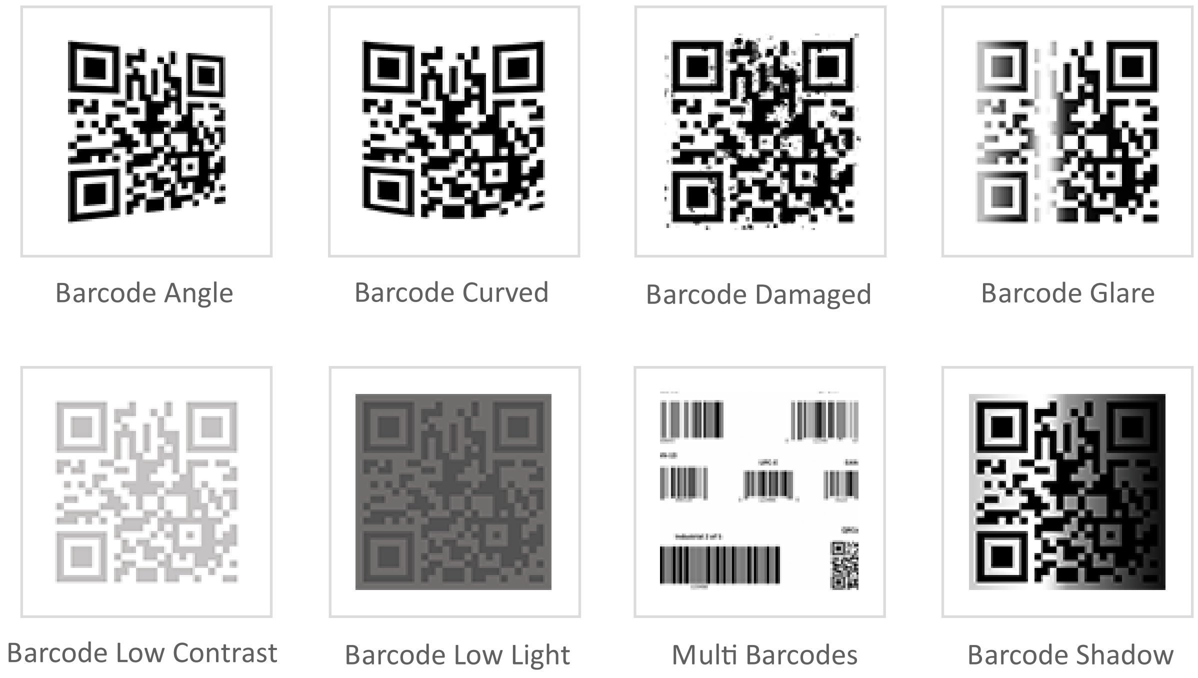
Binarization — converts images to black and white leaving only two levels, allowing for easier distinction between the edges of images.



Despeckle — considers the average of pixels in an area to try and detect whether a pixel is the color it should be or if it is noise created during an image acquisition.

19787772301
Before despeckle
19787772301
After despeckle

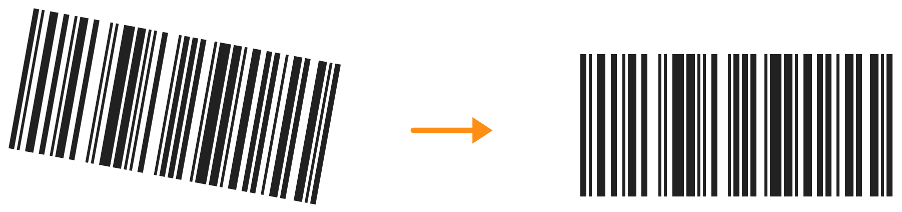
Various Damaged Barcode Scenarios



Deskew A Barcode

To deskew a barcode means to essentially straighten it out. When a barcode is skewed, depending on the angle and how skewed it is, the individual lines of a barcode can appear thicker or thinner than normal. So, when de-skewing occurs, it can actually worsen things.

Also, deskewing almost always reduces the image quality, which can result in reduced barcode recognition accuracy. As with all other obstacles mentioned herein, you can use pre-processing features of an SDK to auto-deskew barcodes while maximizing their recognition accuracy. Regardless, having to deskew will take time and anything that takes time drops speed.



Before deskew

After deskew

Scan Distance

For 1D barcodes, scan distance is important as it can impact barcode recognition in that you may scan unwanted pixels. For example, the number “1” on a page, whether part of a text in a sentence or part of a barcode symbol, may be confused for a bar depending on its location and the scan distance. Thus, it is ideal to set a scan distance of five pixels or more for optimal decoding speed and accuracy. Generally, the higher scan distance (more than five) you can set the more optimal speed will become.

The barcode reader’s viewing angle can also have an impact on the required distance and the reader type you will need. Generally, laser scanners are better at reading at distances more than two feet, such as compared to smartphones. They are also generally better in low-light situations.

With effective pre-processing built-into good software, damaged or hard-to-read barcodes can quickly be corrected during scanning to improve recognition rates.

V. Getting started with Barcodes

A good place to start is with understanding what barcode symbology is most ideal for your application. This might include considering any industry and related regulation needs and requirements. Make sure to follow common practices. Be clear about the character set you need to support for your business. There are various other technical considerations too. Specifically, developers need to weigh the advantages and disadvantages of building versus buying components.

Barcode Support
If you're looking to build a barcode reader application, ultimately, you will likely turn to an SDK to save you on time, resources, and costs. So, you'll need to be sure the SDK can support the barcode technology requirements you have. This means it should have the ability to decode 1D and 2D barcodes. It should also allow decoding from popular sources, such as images, PDF files, or cameras.

Platform Support
You'll want to verify that the SDK can work across platforms. This includes desktop applications you might create in .NET or with popular web browsers if using a web-based approach. Today, you can't ignore mobile applications either. So, consider whether the SDK can capture and scan barcodes from smartphone cameras and webcams, whether a photo or a camera live video stream.

Scan Barcode
Result:
Format: QR_code
Text:www.dynamsoft.com
Mobile APPs
Barcode Online Scanner
Web
Desktop

Programming Language Support
The SDK you choose should also be easy to integrate into your application. Thus, it should support key APIs such as C, C++, .NET, and JavaScript, and 32-bit or 64-bit environments. It should also support widely used programming languages, such as C#, VB.NET, Java, C++, VBScript, JavaScript, Python, Perl, Ruby, Swift, Objective-C, and so on.

Optimization Options

We touched on the importance of being able to customize where on a page a barcode reader should scan. In addition to zonal scan capabilities, a good SDK will let developers set page numbers to scan. This should include allowing a single barcode scan per page or multi-code scans across single or multiple pages. Equally important is the SDK's capability to deal with imperfect barcode scan scenarios with pre-processing. This includes angled barcodes, damaged barcodes, and unreadable barcodes.

Vendor Support
It's sometimes also unavoidable that you will need technical support. So, be sure the SDK vendor provides multiple avenues for getting in touch. It's all too common nowadays to send people to a knowledge base first and then email with a response days or weeks later. So, be sure your vendor of choice also offers more immediate technical support options, such as chat and phone support.

Vendor Updates
Finally, consider how often your SDK vendor updates their solutions. The features you need in a barcode reader SDK today might be different later. So, if your vendor isn't active in improving its products, they may not later support a new feature you need down the road.

Nothing allows you to see just how good an SDK and its vendor is than doing a trial. So, be sure to check out any vendors offering SDK trial opportunities and online demos.

There are a variety of other sources of rich information to use in helping navigate the implementation of barcodes and maximizing their potential. For some of these resources, see below

About Dynamsoft Barcode Reader

Dynamsoft's Barcode Reader SDK enables you to efficiently embed barcode reading functionality in your web, desktop or mobile application using just a few lines of code. This can save you months of added development time and extra costs. With our SDK, you can create high-speed and reliable barcode scanner software to meet your business needs.

Try it for free by downloading the SDK here.

Resources

The Comprehensive Guide to 1D and 2D Barcode Types

https://www.dynamsoft.com/blog/insights/the-comprehensive-guide-to-1d-and-2d-barcodes/

German Red Cross Uses Dynamsoft Barcode Reader SDK to Quickly Automate Barcode-to-PDF Procedure

A healthcare practice uses Dynamsoft's SDK for automation