Dynamsoft™

15 Years of Experience in TWAIN SDKs
and Version Control Solutions

Developer's Guide

Linux Edition

Dbr Dynamsoft Barcode Reader

Table of Contents

Upgrade guide

Developer guide for Linux Edition

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.

This guide shows how to use Dynamsoft Barcode Reader Linux Edition. The core of Linux Edition is written in C and C++ for performance. The library is also wrapped for use with Java, Python, etc.

Supported barcode types

- 1D barcodes: Code 39, Code 93, Code 128, Codabar, ITF, EAN-13, EAN-8, UPC-A, UPC-E, INDUSTRIAL 2 OF 5
- 2D barcodes: QR Code (including Micro QR Code), PDF417 (including Micro PDF417), DataMatrix, Aztec, MaxiCode (mode 2-5), DotCode
- Patch Code
- GS1 DataBar (Omnidirectional, Truncated, Stacked, Stacked Omnidirectional, Limited, Expanded, Expanded Stacked)
- GS1 Composite Code
- Postal Code (USPS Intelligent Mail, POSTNET, PLANET, Australia Post barcode, RM4SCC)

Dynamsoft Barcode Reader editions

Edition	Supported Programing Languages
Windows Edition	C, C++, C#, VB .NET, PHP, Java
Linux Edition	C, C++, Java
JavaScript Edition	JavaScript
iOS Edition	Swift and Objective-C
Android Edition	Java
Mac Edition (Labs project)	C, C++, Java
Raspberry Pi Edition (Labs project)	C, C++, Java

Installation

The Installation section provides information on:

- System requirements
  - Operating systems
  - Languages and environment
- How to obtain the Dynamsoft Barcode Reader SDK

System requirements

Operating systems

- Linux x64 (Ubuntu 14.04.4+ LTS, Debian 8+, etc.)

Note: If you want to support Linux arm, please check out our Raspberry Pi Edition

Languages and environment

Dynamsoft Barcode Reader Linux Edition provides C, C++ and Java APIs.

API	Environment
C, C++	64-bit
Java	JDK 1.7

How to obtain the Dynamsoft Barcode Reader SDK

To install Dynamsoft Barcode Reader Linux Edition on your development machine, you can download the SDK from Dynamsoft website and unpack the .tar.gz file.

After extracting, you can find samples for supported platforms in the **samples** folder.

Build a Hello World application

Prerequisites

Before getting started, please make sure you've already installed Dynamsoft Barcode Reader SDK. If not, please refer to the Installation section to install the development kit on your machine.

To get a free trial license, please refer to the Use a trial key section.

Start coding

This section will show how to build a Hello World application that reads barcodes from a static image in different programming languages.

- Create a C app for barcode reading
- Create a C++ app for barcode reading
- Create a Java app for barcode reading
- Create a Python app for barcode reading
- Create a PHP app for barcode reading

Create a C app for barcode reading

To build a C application that reads barcodes from an image, you can follow the steps below:

1. Create a makefile with the following code:

```
CC=gcc
CCFLAGS=-c
DBRLIB_PATH=<relative path>/lib
LDFLAGS=-L $(DBRLIB_PATH) -Wl,-rpath=$(DBRLIB_PATH) -Wl,-rpath=./
DBRLIB=-lDynamsoftBarcodeReader
TARGET=ReadBarcode
OBJECT=ReadBarcode.o
SOURCE=ReadBarcode.c
# build rule for target.
$(TARGET): $(OBJECT)
$(CC) -o $(TARGET) $(OBJECT) $(STDLIB) $(DBRLIB) $(LDFLAGS)
# target to build an object file
$(OBJECT): $(SOURCE)
$(CC) $(CCFLAGS) $(SOURCE)
# the clean target
.PHONY : clean
clean:
rm -f $(OBJECT) $(TARGET)
```

Note: Please replace with your own relative path in the code.

2. Create an empty `ReadBarcode.c` file and copy the following code to the file.

```
#include <stdio.h>
#include "<relative path>/DynamsoftBarcodeReader.h"
//Please replace <relative path> with your own relative path in the code.
int main()
{
 // Define variables
 void *hBarcode = NULL;
 int iRet = -1;
 int iIndex = 0;
```

```
   int iLicMsg = -1;
   TextResultArray* pResult = NULL;
   hBarcode = DBR_CreateInstance();

   // Initialize license prior to any decoding
   iLicMsg = DBR_InitLicense(hBarcode, "<your license key here>");

   //If error occurs to the license initialization
    if (iLicMsg != DBR_OK)
   {
       printf("Failed to initialize the license successfully: %d\r\n%s\r\n", iLicMsg, DBR_GetErrorString(iLicMs
g));
       return iLicMsg;
   }

   // Started decoding
   DBR_DecodeFile(hBarcode, "<your image file full path>", "");

   // Get the text result
   iRet = DBR_GetAllTextResults(hBarcode, &pResult);

   // If error occurs
   if (iRet != DBR_OK)
   {
       printf("Failed to read barcode: %d\r\n%s\r\n", iRet, DBR_GetErrorString(iRet));
       return iRet;
   }

   // If succeeds
   printf("%d total barcode(s) found. \n", pResult->resultsCount);
   for (iIndex = 0; iIndex < pResult->resultsCount; iIndex++)
   {
       printf("Result %d\n", iIndex + 1);
       printf("Barcode Format: %s\n", pResult->results[iIndex]->barcodeFormatString);
       printf("Text reads: %s \n", pResult->results[iIndex]->barcodeText);
   }

   // Finally release BarcodeResultArray and destroy instance
   DBR_FreeTextResults(&pResult);
   DBR_DestroyInstance(hBarcode);
   system("pause");
   return 0;
}
```

3. Run the project.

   Open the Terminal, use command `cd` to change the current directory to the one where the makefile resides, and use command `make` to build the executable program.

To deploy your application, make sure all the `.so` files are in the same folder as the execution file. See the Distribution section for more details.

Create a C++ app for barcode reading

To build a C++ application that reads barcodes from an image, you can follow the steps below:

1. Create a makefile with the following code.

```
CC=gcc
CCFLAGS=-c
DBRLIB_PATH=<relative path>/lib
LDFLAGS=-L $(DBRLIB_PATH) -Wl,-rpath=$(DBRLIB_PATH) -Wl,-rpath=./
```

```
DBRLIB=-lDynamsoftBarcodeReader
STDLIB=-lstdc++
TARGET=ReadBarcode
OBJECT=ReadBarcode.o
SOURCE=ReadBarcode.cpp
# build rule for target.
$(TARGET): $(OBJECT)
$(CC) -o $(TARGET) $(OBJECT) $(STDLIB) $(DBRLIB) $(LDFLAGS)
# target to build an object file
$(OBJECT): $(SOURCE)
$(CC) $(CCFLAGS) $(SOURCE)
# the clean target
.PHONY : clean
clean:
rm -f $(OBJECT) $(TARGET)
```

Note: Please replace with your own relative path in the code.

2. Create an empty `ReadBarcode.cpp` file and copy the following code to the file.

```
#include <stdio.h>
#include "<relative path>/DynamsoftBarcodeReader.h"
int main()
{
 // Define variables
 int iRet = -1;
 int iLicMsg = -1;
 TextResultArray *paryResult = NULL;

 // Initialize license prior to any decoding
 CBarcodeReader reader;
 iLicMsg = reader.InitLicense("<your license key here>");

 //If error occurs to the license initialization
 if (iLicMsg != DBR_OK)
 {
     printf("Failed to initialize the license successfully: %d\r\n%s\r\n", iLicMsg, DBR_GetErrorString(iLicMs
g));
     return iLicMsg;
 }

 // Start decoding
 iRet = reader.DecodeFile("<your image file full path>", "");

 // If error occurs
 if (iRet != DBR_OK)
 {
     printf("Failed to read barcode: %d\r\n%s\r\n", iRet, DBR_GetErrorString(iRet));
     return iRet;
 }

 // If succeeds
 reader.GetAllTextResults(&paryResult);
 printf("%d total barcodes found. \r\n", paryResult->resultsCount);
 for (int iIndex = 0; iIndex < paryResult->resultsCount; iIndex++)
 {
     printf("Result %d\r\n", iIndex + 1);
     printf("BarcodeFormat: %s\r\n", paryResult->results[iIndex]->barcodeFormatString);
     printf("Text read: %s\r\n", paryResult->results[iIndex]->barcodeText);
 }

 // Finally release BarcodeResultArray
```

```
  CBarcodeReader::FreeTextResults(&paryResult);
  system("pause");
  return 0;
}
```

Note: Please update the , and with valid values respectively in the code.

3. Run the project.

   Open the Terminal, use command `cd` to change the current directory to the one where the makefile resides, and use command `make` to build the executable program.

To deploy your application, make sure all the `.so` files are in the same folder as the execution file. See the Distribution section for more details.

Create a Java app for barcode reading

To build a Java application that reads barcodes from an image, you can follow the steps below:

1. Open Eclipse and create a new Java project `HelloDBR` .
2. Add the required JAR file to your project.

   Click **File** > **Properties** > **Java Build Path** > **Libraries** > **Add external JARs**, add `DynamsoftBarcodeReader_{edition}.jar` and click **Apply**.

   The JAR file can be found at `[INSTALLATION FOLDER]\Components\Java\` .
3. Import the header.

```
import com.dynamsoft.barcode.*;
```

4. Replace the code of `HelloDBR` with the following.

   Please update `<your image file full path>` and `<your license key here>` in the code accordingly.

```
public class HelloDBR {
    public static void main(String[] args) {
        try {
            BarcodeReader dbr = new BarcodeReader();
            dbr.initLicense("<Put your license key here>");
            TextResult[] result = dbr.decodeFile("<your image file full path>", "");
            String output = "";
            for(int i =0; i<result.length;i++){
                output += "Barcode Text: ";
                output += result[i].barcodeText + "\n\n";
            }
            System.out.println(output);
        }
        catch(Exception ex) {
        }
    }
}
```

5. Run the project.

Create a Python app for barcode reading

You can easily create a Python extension with the C/C++ API of Dynamsoft Barcode Reader.

For more information on how to create a Python extension for barcode reading, please refer to DBR Python Extension .

Create a PHP app for barcode reading

Dynamsoft Barcode Reader provides a PHP wrapper for Linux. With it, you can easily implement barcode reading in your PHP web applications.

For more information on how to create a PHP application for barcode scanning, Please refer to Use Dynamsoft Barcode Reader to Build a Barcode Scanner in PHP on Linux.

Decoding methods

The SDK provides multiple decoding methods that support reading barcodes from different sources, including static images, video stream, files in memory, base64 string, bitmap, etc. Here is a list of all decoding methods:

- DecodeFile: Reads barcodes from a specified file (BMP, JPEG, PNG, GIF, TIFF or PDF).
- DecodeBase64String: Reads barcodes from a base64 encoded string of a file.
- DecodeBitmap: Reads barcodes from a bitmap. When handling multi-page images, it will only decode the current page.
- DecodeBuffer: Reads barcodes from raw buffer.
- DecodeFileInMemory: Decodes barcodes from an image file in memory.

Code snippets are provided in C/C++ as examples below. Not all supported programming languages are covered in this guide. You can find more samples in more programming languages at Code Gallery or Github Repositories.

DecodeFile

Reads barcodes from a statc image file.

Code snippet in c:

```c
void *hBarcode = NULL;
TextResultArray* pResult = NULL;
hBarcode = DBR_CreateInstance();
// Initialize license prior to any decoding
//Replace “Put your license key here” with your own license
DBR_InitLicense(hBarcode, "<Put your license key here>");
//Replace “Put the path of your file here” with your own file path
DBR_DecodeFile(hBarcode,"<Put your file path here>","");
DBR_GetAllTextResults(hBarcode, &pResult);
DBR_FreeTextResults(&pResult);
DBR_DestroyInstance(hBarcode);
```

Code snippet in cpp:

```cpp
char sError[512];
TextResultArray* paryResult = NULL;
CBarcodeReader* reader = new CBarcodeReader();
//Initialize license prior to any decoding
//Replace "<Put your license key here>" with your own license
reader->InitLicense("<Put your license key here>");
//Replace “Put the path of your file here” with your own file path
reader->DecodeFile("<Put your file path here>", "");
// If succeeds
reader->GetAllTextResults(&paryResult);
CBarcodeReader::FreeTextResults(&paryResult);
delete reader;
```

DecodeBase64String

Reads barcodes from the base64 encoded string of a file.

Code snippet in c:

```c
void* hBarcode = DBR_CreateInstance();
int iRet = -1;
int iIndex = 0;
int iLicMsg = -1;
unsigned char* pBufferBytes;
int nFileSize = 0;
char* strBase64String;
TextResultArray* pResult = NULL;

// Replace "Put your license key here" with your own trial license
iLicMsg = DBR_InitLicense(hBarcode, "Put your license key here");

// If error occurs to the license
if (iLicMsg != DBR_OK) {
   printf("Failed to initialize the license successfully: %d\r\n%s\r\n", iLicMsg, DBR_GetErrorString(iLicMsg));
   return iLicMsg;
}
// Replace "Put the path of your file here" with your own file path
GetFileStream("Put the path of your file here", &pBufferBytes, &nFileSize);
GetFileBase64String(pBufferBytes, &strBase64String);
DBR_DecodeBase64String(hBarcode, strBase64String, "");
iRet = DBR_GetAllTextResults(hBarcode, &pResult);
// If error occurs
if (iRet != DBR_OK) {
   printf("Failed to read barcode: %d\r\n%s\r\n", iRet, DBR_GetErrorString(iRet));
   return iRet;
}
printf("%d total barcode(s) found. \n", pResult->resultsCount);
for (iIndex = 0; iIndex < pResult->resultsCount; iIndex++)
{
   printf("Result %d\n", iIndex + 1);
   printf("Barcode Format: %s\n", pResult->results[iIndex]->barcodeFormatString);
   printf("Text reads: %s \n", pResult->results[iIndex]->barcodeText);
}
DBR_FreeTextResults(&pResult);
DBR_DestroyInstance(hBarcode);
return 0;
```

Code snippet in cpp:

```cpp
int main()
{
   int iRet = -1;
   int iLicMsg = -1;
   char* strBase64String;
   TextResultArray* paryResult = NULL;
   PublicRuntimeSettings* runtimeSettings = new PublicRuntimeSettings();
   CBarcodeReader* reader = new CBarcodeReader();
   // Initialize license prior to any decoding
   // Replace "Put your license key here" with your own license
   iLicMsg = reader->reader.InitLicense("Put your license key here ");
   // If error occurs to the license
   if (iLicMsg != DBR_OK)
   {
      printf("Failed to initialize the license successfully: %d\r\n%s\r\n", iLicMsg, CBarcodeReader::GetErrorStri
ng(iLicMsg));
      return iLicMsg;
   }
   // Replace "Put the path of your file here" with your own file path
   GetFileStream("Put the path of your file here ", &pBufferBytes, &nFileSize);
   GetFileBase64String(pBufferBytes, &strBase64String);
   iRet = reader->DecodeBase64String(strBase64String, "");
   // If error occurs
   if (iRet != DBR_OK)
   {
      printf("Failed to read barcode: %d\r\n%s\r\n", iRet, CBarcodeReader::GetErrorString(iRet));
      return iRet;
   }
   // If succeeds
   reader->GetAllTextResults(&paryResult);
   printf("%d total barcodes found. \r\n", paryResult->resultsCount);
   for (int iIndex = 0; iIndex < paryResult->resultsCount; iIndex++)
   {
      printf("Result %d\r\n", iIndex + 1);
      printf("BarcodeFormat: %s\r\n", paryResult->results[iIndex]->barcodeFormatString);
      printf("Text read: %s\r\n", paryResult->results[iIndex]->barcodeText);
   }
   CBarcodeReader::FreeTextResults(&paryResult);
   delete runtimeSettings;
   delete reader;
   getchar();
   return 0;
}
```

DecodeBitmap

`DecodeBitmap()` reads barcodes from a bitmap. It will only decode the current page when handling multi-page images.

Code snippet in c:

```
void* hBarcode = DBR_CreateInstance();
HANDLE pDIB;
int iRet = -1;
int iIndex = 0;
int iLicMsg = -1;
TextResultArray* pResult = NULL;
// Replace "Put your license key here" with your own license
iLicMsg = DBR_InitLicense(hBarcode, "Put your license key here");
// If error occurs to the license
if (iLicMsg != DBR_OK) {
    printf("Failed to initialize the license successfully: %d\r\n%s\r\n", iLicMsg, DBR_GetErrorString(iLicMsg));
    return iLicMsg;
}
// Replace "Put the path of your file here" with your own file path
GetDIBFromImage("Put the path of your file here ", &pDIB);
DBR_DecodeDIB(hBarcode, pDIB, "");
iRet = DBR_GetAllTextResults(hBarcode, &pResult);
// If error occurs
if (iRet != DBR_OK) {
    printf("Failed to read barcode: %d\r\n%s\r\n", iRet, DBR_GetErrorString(iRet));
    return iRet;
}
printf("%d total barcode(s) found. \n", pResult->resultsCount);
for (iIndex = 0; iIndex < pResult->resultsCount; iIndex++)
{
    printf("Result %d\n", iIndex + 1);
    printf("Barcode Format: %s\n", pResult->results[iIndex]->barcodeFormatString);
    printf("Text reads: %s \n", pResult->results[iIndex]->barcodeText);
}
DBR_FreeTextResults(&pResult);
DBR_DestroyInstance(hBarcode);
return 0;
```

Code snippet in cpp:

```
int main()
```

```
{
   int iRet = -1;
   int iLicMsg = -1;
   HANDLE pDIB;
   TextResultArray* paryResult = NULL;
   PublicRuntimeSettings* runtimeSettings = new PublicRuntimeSettings();
   CBarcodeReader* reader = new CBarcodeReader();
   // Initialize license prior to any decoding
   //Replace "Put your license key here" with your own license
   iLicMsg = reader->reader.InitLicense("Put your license key here ");
   // If error occurs to the license
   if (iLicMsg != DBR_OK)
   {
      printf("Failed to initialize the license successfully: %d\r\n%s\r\n", iLicMsg, CBarcodeReader::GetErrorStri
ng(iLicMsg));
      return iLicMsg;
   }
   // Replace "Put the path of your file here" with your own file path
   GetDIBFromImage("Put the path of your file here ", &pDIB);
   iRet = reader->DecodeDIB(pDIB, "");
   // If error occurs
   if (iRet != DBR_OK)
   {
      printf("Failed to read barcode: %d\r\n%s\r\n", iRet, CBarcodeReader::GetErrorString(iRet));
      return iRet;
   }
   // If succeeds
   reader->GetAllTextResults(&paryResult);
   printf("%d total barcodes found. \r\n", paryResult->resultsCount);
   for (int iIndex = 0; iIndex < paryResult->resultsCount; iIndex++)
   {
      printf("Result %d\r\n", iIndex + 1);
      printf("BarcodeFormat: %s\r\n", paryResult->results[iIndex]->barcodeFormatString);
      printf("Text read: %s\r\n", paryResult->results[iIndex]->barcodeText);
   }
   CBarcodeReader::FreeTextResults(&paryResult);
   delete runtimeSettings;
   delete reader;
   getchar();
   return 0;
}
```

DecodeBuffer

Reads barcodes from raw buffer.

Code snippet in c:

```
void* hBarcode = DBR_CreateInstance();
int iRet = -1;
int iIndex = 0;
int iLicMsg = -1;
unsigned char* pBufferBytes;
int iWidth = 0;
int iHeight = 0;
int iStride = 0;
ImagePixelFormat format;
TextResultArray* pResult = NULL;
// Replace "Put your license key here" with your own license
iLicMsg = DBR_InitLicense(hBarcode, "Put your license key here");
// If error occurs to the license
if (iLicMsg != DBR_OK) {
    printf("Failed to initialize the license successfully: %d\r\n%s\r\n", iLicMsg, DBR_GetErrorString(iLicMsg));
    return iLicMsg;
}
// Replace "Put the path of your file here" with your own file path
GetBufferFromFile("Put the path of your file here ", &pBufferBytes, &iWidth, &iHeight, &iStride, &format);
DBR_DecodeBuffer(hBarcode, pBufferBytes, iWidth, iHeight, iStride, format, "");
iRet = DBR_GetAllTextResults(hBarcode, &pResult);
// If error occurs
if (iRet != DBR_OK) {
    printf("Failed to read barcode: %d\r\n%s\r\n", iRet, DBR_GetErrorString(iRet));
    return iRet;
}
printf("%d total barcode(s) found. \n", pResult->resultsCount);
for (iIndex = 0; iIndex < pResult->resultsCount; iIndex++)
{
    printf("Result %d\n", iIndex + 1);
    printf("Barcode Format: %s\n", pResult->results[iIndex]->barcodeFormatString);
    printf("Text reads: %s \n", pResult->results[iIndex]->barcodeText);
}
DBR_FreeTextResults(&pResult);
DBR_DestroyInstance(hBarcode);
return 0;
```

Code snippet in cpp:

```
int main()
{
    int iRet = -1;
    int iLicMsg = -1;
    int iWidth = 0;
    int iHeight = 0;
    int iStride = 0;
    unsigned char* pBufferBytes;
    ImagePixelFormat format;
    TextResultArray* paryResult = NULL;
    PublicRuntimeSettings* runtimeSettings = new PublicRuntimeSettings();
    CBarcodeReader* reader = new CBarcodeReader();
    // Initialize license prior to any decoding
    // Replace "Put your license key here" with your own license
    iLicMsg = reader->reader.InitLicense("Put your license key here ");
    // If error occurs to the license
    if (iLicMsg != DBR_OK)
    {
        printf("Failed to initialize the license successfully: %d\r\n%s\r\n", iLicMsg, CBarcodeReader::GetErrorStri
ng(iLicMsg));
        return iLicMsg;
    }
    // Replace "Put the path of your file here" with your own file path
    GetBufferFromFile("Put the path of your file here ", &pBufferBytes, &iWidth, &iHeight, &iStride, &format);
    iRet = reader->DecodeBuffer(pBufferBytes, iWidth, iHeight, iStride, format, "");
    if (iRet != DBR_OK)
    {
        printf("Failed to read barcode: %d\r\n%s\r\n", iRet, CBarcodeReader::GetErrorString(iRet));
        return iRet;
    }
    // If succeeds
    reader->GetAllTextResults(&paryResult);
    printf("%d total barcodes found. \r\n", paryResult->resultsCount);
    for (int iIndex = 0; iIndex < paryResult->resultsCount; iIndex++)
    {
        printf("Result %d\r\n", iIndex + 1);
        printf("BarcodeFormat: %s\r\n", paryResult->results[iIndex]->barcodeFormatString);
        printf("Text read: %s\r\n", paryResult->results[iIndex]->barcodeText);
    }
    CBarcodeReader::FreeTextResults(&paryResult);
    delete runtimeSettings;
    delete reader;
    getchar();
    return 0;
}
```

DecodeFileInMemory

Decodes barcodes from an image file in memory.

Code snippet in c:

```
void* hBarcode = DBR_CreateInstance();
int iRet = -1;
int iIndex = 0;
int iLicMsg = -1;
unsigned char* pFileBytes;
int nFileSize = 0;
TextResultArray* pResult = NULL;
// Replace "Put your license key here" with your own license
iLicMsg = DBR_InitLicense(hBarcode, "Put your license key here");
// If error occurs to the license
if (iLicMsg != DBR_OK) {
    printf("Failed to initialize the license successfully: %d\r\n%s\r\n", iLicMsg, DBR_GetErrorString(iLicMsg));
    return iLicMsg;
}
// Replace "Put the path of your file here" with your own file path
GetFileStream("Put the path of your file here ", &pFileBytes, &nFileSize);
DBR_DecodeFileInMemory(hBarcode, pFileBytes, nFileSize, "");

iRet = DBR_GetAllTextResults(hBarcode, &pResult);

// If error occurs
if (iRet != DBR_OK) {
    printf("Failed to read barcode: %d\r\n%s\r\n", iRet, DBR_GetErrorString(iRet));
    return iRet;
}
printf("%d total barcode(s) found. \n", pResult->resultsCount);
for (iIndex = 0; iIndex < pResult->resultsCount; iIndex++)
{
    printf("Result %d\n", iIndex + 1);
    printf("Barcode Format: %s\n", pResult->results[iIndex]->barcodeFormatString);
    printf("Text reads: %s \n", pResult->results[iIndex]->barcodeText);
}
DBR_FreeTextResults(&pResult);
DBR_DestroyInstance(hBarcode);
return 0;
```

Code snippet in cpp:

```
int main()
{
    int iRet = -1;
    int iLicMsg = -1;
    int nFileSize = 0;
    unsigned char* pFileBytes;
    TextResultArray* paryResult = NULL;
    PublicRuntimeSettings* runtimeSettings = new PublicRuntimeSettings();
    CBarcodeReader* reader = new CBarcodeReader();
    // Initialize license prior to any decoding
    // Replace "Put your license key here" with your own license
    iLicMsg = reader->InitLicense("Put your license key here ");
    // Replace "Put the path of your file here" with your own file path
    GetFileStream("Put the path of your file here ", &pFileBytes, &nFileSize);
    iRet = reader->DecodeFileInMemory(pFileBytes, nFileSize, "");
    // If error occurs
    if (iRet != DBR_OK)
    {
       printf("Failed to read barcode: %d\r\n%s\r\n", iRet, CBarcodeReader::GetErrorString(iRet));
       return iRet;
    }
    // If succeeds
    reader->GetAllTextResults(&paryResult);
    printf("%d total barcodes found. \r\n", paryResult->resultsCount);
    for (int iIndex = 0; iIndex < paryResult->resultsCount; iIndex++)
    {
       printf("Result %d\r\n", iIndex + 1);
       printf("BarcodeFormat: %s\r\n", paryResult->results[iIndex]->barcodeFormatString);
       printf("Text read: %s\r\n", paryResult->results[iIndex]->barcodeText);
    }
    CBarcodeReader::FreeTextResults(&paryResult);
    delete runtimeSettings;
    delete reader;
    getchar();
    return 0;
}
```

Barcode reading settings

Calling the decoding methods directly will use the default scanning modes and it will satisfy most of the needs. The SDK also allows you to adjust the scanning settings to optimize the scanning performance for different usage scenarios.

There are two ways to change the barcode reading settings - using the `PublicRuntimeSettings` Struct or template. For new developers, We recommend you to start with the `PublicRuntimeSettings` struct; For those who are experienced with the SDK, you may use a template which is more flexible and easier to update.

- Use PublicRuntimeSettings Struct to Change Settings
- Use A Template to Change Settings

Use PublicRuntimeSettings struct to change settings

Here are some common scanning settings you might find helpful:

- Specify barcode type to read
- Specify maximum barcode count
- Specify a scan region

For more scanning settings guide, check out the How To section.

Learn more about PublicRuntimeSettings Struct.

Specify which barcode type to read

By default, the SDK will read all the supported barcode formats from the image. (See Product Overview for the full supported barcode list.)

If your full license only covers some barcode formats, you can use `barcodeFormatIds` to specify the barcode format(s).

For example, to enable only 1D barcode reading, you can use the following code:

Code snippet in c:

```c
void *hBarcode = NULL;
char sError[512];
TextResultArray* pResult = NULL;
PublicRuntimeSettings runtimeSettings;
hBarcode = DBR_CreateInstance();

// Initialize license prior to any decoding
//Replace "<Put your license key here>" with your own license
DBR_InitLicense(hBarcode, "<Put your license key here>");
//Set the barcode format
DBR_GetRuntimeSettings(hBarcode, &runtimeSettings);
runtimeSettings.barcodeFormatIds = 2047; // OneD barcode
DBR_UpdateRuntimeSettings(hBarcode, &runtimeSettings,sError,512);

//Replace "Put the path of your file here" with your own file path
DBR_DecodeFile(hBarcode,"<Put your file path here>","");
DBR_GetAllTextResults(hBarcode, &pResult);
DBR_FreeTextResults(&pResult);
DBR_DestroyInstance(hBarcode);
```

Code snippet in cpp:

```cpp
char sError[512];
```

```
TextResultArray* paryResult = NULL;
PublicRuntimeSettings* runtimeSettings = new PublicRuntimeSettings();
CBarcodeReader* reader = new CBarcodeReader();
//Initialize license prior to any decoding
//Replace "<Put your license key here>" with your own license
reader->InitLicense("<Put your license key here>");

//Set the barcode format
reader->GetRuntimeSettings(runtimeSettings);
runtimeSettings->barcodeFormatIds = 2047; //OneD barcode
reader->UpdateRuntimeSettings(runtimeSettings, sError, 512);

//Replace "<Put the path of your file here>" with your own file path
reader->DecodeFile("<Put your file path here>", "");
// If succeeds
reader->GetAllTextResults(&paryResult);
CBarcodeReader::FreeTextResults(&paryResult);
delete runtimeSettings;
delete reader;
```

Code snippet in Java:

```
BarcodeReader dbr = new BarcodeReader();

// set barcodeFromatIds via PublicRuntimeSettings instance and update it to BarcodeReader instance
PublicRuntimeSettings rts = dbr.getRuntimeSettings();
rts.barcodeFormatIds =  0x7FF;// OneD barcode
dbr.updateRuntimeSettings(rts);

// Initialize license prior to any decoding
//Replace "<Put your license key here>" with your own license
dbr.initLicense("<Put your license key here>");
//Replace "Put the path of your file here" with your own file path
TextResult[] result = dbr.decodeFile("<Put your file path here>","");
```

Specify maximum barcode count

By default, the SDK will read as many barcodes as it can. To increase the recognition efficiency, you can use `expectedBarcodesCount` to specify the maximum number of barcodes to recognize according to your scenario.

Code snippet in c:

```
void *hBarcode = NULL;
char sError[512];
TextResultArray* pResult = NULL;
PublicRuntimeSettings runtimeSettings;
hBarcode = DBR_CreateInstance();
// Initialize license prior to any decoding
//Replace "<Put your license key here>" with your own license
DBR_InitLicense(hBarcode, "<Put your license key here>");

//Set the number of barcodes to be expected
DBR_GetRuntimeSettings(hBarcode, &runtimeSettings);
runtimeSettings.expectedBarcodesCount = 1;
DBR_UpdateRuntimeSettings(hBarcode, &runtimeSettings, sError, 512);

//Replace "<Put the path of your file here>" with your own file path
DBR_DecodeFile(hBarcode,"<Put your file path here>","");
DBR_GetAllTextResults(hBarcode, &pResult);
```

```
DBR_FreeTextResults(&pResult);
DBR_DestroyInstance(hBarcode);
```

Code snippet in cpp:

```cpp
char sError[512];
TextResultArray* paryResult = NULL;
PublicRuntimeSettings* runtimeSettings = new PublicRuntimeSettings();
CBarcodeReader* reader = new CBarcodeReader();
//Initialize license prior to any decoding
//Replace "<Put your license key here>" with your own license
reader->InitLicense("<Put your license key here>");

//Set the number of barcodes to be expected
reader->GetRuntimeSettings(runtimeSettings);
runtimeSettings->expectedBarcodesCount = 1;
reader->UpdateRuntimeSettings(runtimeSettings, sError, 512);

//Replace "Put the path of your file here" with your own file path
reader->DecodeFile("<Put your file path here>", "");
// If succeeds
reader->GetAllTextResults(&paryResult);
CBarcodeReader::FreeTextResults(&paryResult);
delete runtimeSettings;
delete reader;
```

Specify a scan region

By default, the barcode reader will search the whole image for barcodes. This can lead to poor performance especially when dealing with high-resolution images. You can speed up the recognition process by restricting the scanning region.

To specify a region, you will need to define an area. The following code shows how to create a template string and define the region.

Code snippet in c:

```c
void *hBarcode = NULL;
char sError[512];
TextResultArray* pResult = NULL;
PublicRuntimeSettings runtimeSettings;
hBarcode = DBR_CreateInstance();
// Initialize license prior to any decoding
//Replace "<Put your license key here>" with your own license
DBR_InitLicense(hBarcode, "<Put your license key here>");

//Decode the barcodes on the left half of the image
DBR_GetRuntimeSettings(hBarcode, &runtimeSettings);
runtimeSettings.region.regionBottom = 100;
runtimeSettings.region.regionLeft = 0;
runtimeSettings.region.regionRight = 50;
runtimeSettings.region.regionTop = 0;
runtimeSettings.region.regionMeasuredByPercentage = 1; //The region is determined by percentage
DBR_UpdateRuntimeSettings(hBarcode, &runtimeSettings,sError,512);

//Replace "<Put the path of your file here>" with your own file path
DBR_DecodeFile(hBarcode,"put your file path here","");
DBR_GetAllTextResults(hBarcode, &pResult);
DBR_FreeTextResults(&pResult);
DBR_DestroyInstance(hBarcode);
```

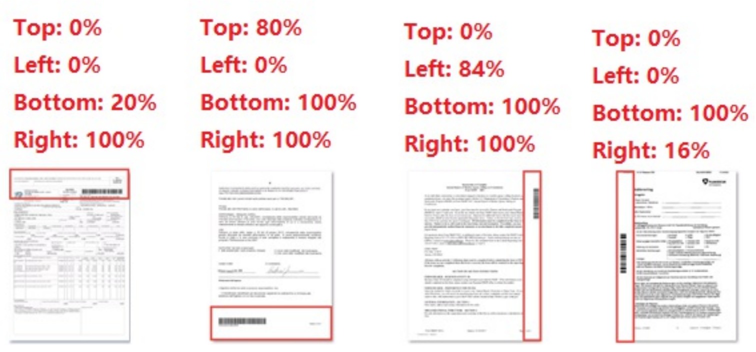
Code snippet in cpp:

```cpp
char sError[512];
TextResultArray* paryResult = NULL;
PublicRuntimeSettings* runtimeSettings = new PublicRuntimeSettings();
CBarcodeReader* reader = new CBarcodeReader();
//Initialize license prior to any decoding
//Replace "<Put your license key here>" with your own license
reader->InitLicense("Put your license key here");

//Decode the barcodes on the left of the image
reader->GetRuntimeSettings(runtimeSettings);
runtimeSettings->region.regionBottom = 100;
runtimeSettings->region.regionLeft = 0;
runtimeSettings->region.regionRight = 50;
runtimeSettings->region.regionTop = 0;
runtimeSettings->region.regionMeasuredByPercentage = 1; //The region is determined by percentage
reader->UpdateRuntimeSettings(runtimeSettings, sError, 512);

//Replace "<Put the path of your file here>" with your own file path
reader->DecodeFile("<Put your file path here>", "");
// If succeeds
reader->GetAllTextResults(&paryResult);
CBarcodeReader::FreeTextResults(&paryResult);
delete runtimeSettings;
delete reader;
```

For example:



Use a template to change settings

Besides the option of using the `PublicRuntimeSettings` struct, the SDK also provides `InitRuntimeSettingsWithString()` and `InitRuntimeSettingsWithFile()` APIs that enable you to use a template to control all the runtime settings. With a template, instead of writing many codes to modify the settings, you can manage all the runtime settings in a JSON file/string.

Code snippet in c:

```c
void *hBarcode = NULL;
char sError[512];
TextResultArray* pResult = NULL;
hBarcode = DBR_CreateInstance();
// Initialize license prior to any decoding
//Replace "<Put your license key here>" with your own license
DBR_InitLicense(hBarcode, "<Put your license key here>");

//Use a template to modify the runtime settings
//DBR_InitRuntimeSettingsWithString() can also be used to modify the runtime settings with a json string
```

```
DBR_InitRuntimeSettingsWithFile(hBarcode, "<Put your file path here>", CM_OVERWRITE, sError, 512);

//Output runtime settings to a json file.
//DBR_OutputLicenseToString() can also be used to output the settings to a string
DBR_OutputSettingsToFile(hBarcode, "<Put your file path here>", "runtimeSettings");

//Replace "<Put your file path here>" with your own file path
DBR_DecodeFile(hBarcode,"put your file path here","");
DBR_GetAllTextResults(hBarcode, &pResult);
DBR_FreeTextResults(&pResult);
DBR_DestroyInstance(hBarcode);
```

Code snippet in cpp:

```cpp
char sError[512];
TextResultArray* paryResult = NULL;
CBarcodeReader* reader = new CBarcodeReader();
//Initialize license prior to any decoding
//Replace "Put your license key here" with your own license
reader->InitLicense("Put your license key here");

//Use a template to modify the runtime settings
//InitRuntimeSettingsWithString() can also be used to modify the runtime settings with a json string
reader->InitRuntimeSettingsWithFile("<Put your file path here>", CM_OVERWRITE, sError, 512);

//Output runtime settings to a json file.
//OutputSettingsToString() can also be used to output the settings to a string
reader->OutputSettingsToFile("<Put your file path here>","currentruntime");

//Replace "Put the path of your file here" with your own file path
reader->DecodeFile("Put your file path here", "");
// If succeeds
reader->GetAllTextResults(&paryResult);
CBarcodeReader::FreeTextResults(&paryResult);
delete reader;
```

Below is a template for your reference. To learn more about the APIs, you can check out PublicRuntimeSettings Struct.

```json
{
   "ImageParameter" : {
      "BarcodeFormatIds" : [ "BF_ALL" ],
      "BinarizationModes" : [
         {
            "BlockSizeX" : 0,
            "BlockSizeY" : 0,
            "EnableFillBinaryVacancy" : 1,
            "ImagePreprocessingModesIndex" : -1,
            "Mode" : "BM_LOCAL_BLOCK",
            "ThreshValueCoefficient" : 10
         }
      ],
      "DeblurLevel" : 9,
      "Description" : "",
      "ExpectedBarcodesCount" : 0,
      "GrayscaleTransformationModes" : [
         {
            "Mode" : "GTM_ORIGINAL"
         }
      ],
      "ImagePreprocessingModes" : [
         {
            "Mode" : "IPM_GENERAL"
         }
      ],
      "IntermediateResultSavingMode" : {
         "Mode" : "IRSM_MEMORY"
      },
      "IntermediateResultTypes" : [ "IRT_NO_RESULT" ],
      "MaxAlgorithmThreadCount" : 4,
      "Name" : "runtimesettings",
      "PDFRasterDPI" : 300,
      "Pages" : "",
      "RegionDefinitionNameArray" : null,
      "RegionPredetectionModes" : [
         {
            "Mode" : "RPM_GENERAL"
         }
      ],
      "ResultCoordinateType" : "RCT_PIXEL",
      "ScaleDownThreshold" : 2300,
      "TerminatePhase" : "TP_BARCODE_RECOGNIZED",
      "TextFilterModes" : [
         {
            "MinImageDimension" : 65536,
            "Mode" : "TFM_GENERAL_CONTOUR",
            "Sensitivity" : 0
         }
      ],
      "TextResultOrderModes" : [
         {
            "Mode" : "TROM_CONFIDENCE"
         },
         {
            "Mode" : "TROM_POSITION"
         },
         {
            "Mode" : "TROM_FORMAT"
         }
      ],
      "TextureDetectionModes" : [
         {
            "Mode" : "TDM_GENERAL_WIDTH_CONCENTRATION",
            "Sensitivity" : 5
         }
      ],
      "Timeout" : 10000
   },
   "Version" : "3.0"
}
```

If you want to generate a custom barcode reading template, please refer to this article.

How-to guides

This section covers the following topics:

How to read barcodes from camera stream

Since v7.0, Dynamsoft Barcode Reader SDK added `StartFrameDecoding()` and `StopFrameDecoding()` APIs. With the APIs and OpenCV library, you can easily read barcodes from video stream.

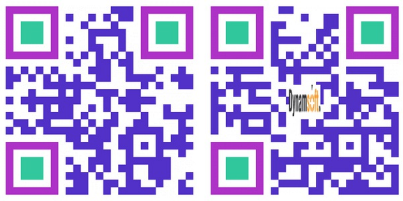
Get a sample: C++ Decode Video Frame >

Note: The sample is for Windows Edition, but the C++ code is the same as the Linux Edition. You can refer to the source code for reference and build your own application.

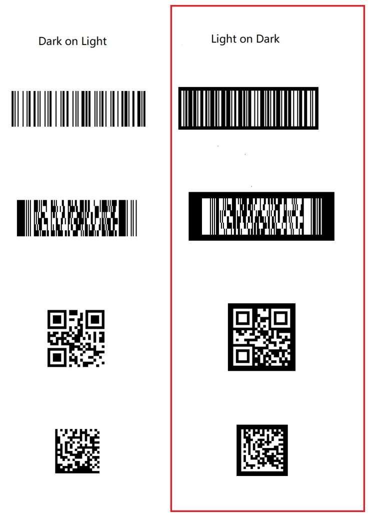
To learn more about the `StartFrameDecoding()` and `StopFrameDecoding()` APIs, see C++ API Reference.

How to read barcodes with different colors

Common barcodes are black and white. However, some barcodes could be in different colors such as the below left barcode. Also, in some cases, a QR code may appear with an image or a logo inside as shown below right. Dynamsoft can decode such special barcodes as well.



In some other cases, the barcodes can be white with a dark background color as shown on the right part of the image below:



To decode such kind of barcodes, the default runtime settings may fail. You can adjust the `PublicRuntimeSettings` like below:

Code snippet in c:

```c
void *hBarcode = NULL;
char sError[512];
TextResultArray* pResult = NULL;
PublicRuntimeSettings runtimeSettings;
hBarcode = DBR_CreateInstance();
// Initialize license prior to any decoding
//Replace "<Put your license key here>" with your own license
DBR_InitLicense(hBarcode, "<Put your license key here>");

//Change the runtime settings to read both normal and inverted barcodes
DBR_GetRuntimeSettings(hBarcode, &runtimeSettings);
runtimeSettings.furtherModes.grayscaleTransformationModes[0] = GTM_ORIGINAL;
runtimeSettings.furtherModes.grayscaleTransformationModes[1] = GTM_INVERTED;
DBR_UpdateRuntimeSettings(hBarcode, &runtimeSettings,sError,512);

//Replace "<Put the path of your file here>" with your own file path
DBR_DecodeFile(hBarcode,"<Put your file path here>","");
DBR_GetAllTextResults(hBarcode, &pResult);
DBR_FreeTextResults(&pResult);
DBR_DestroyInstance(hBarcode);
```

Code snippet in cpp:

```cpp
char sError[512];
TextResultArray* paryResult = NULL;
PublicRuntimeSettings* runtimeSettings = new PublicRuntimeSettings();
CBarcodeReader* reader = new CBarcodeReader();
//Initialize license prior to any decoding
//Replace "<Put your license key here>" with your own license
reader->InitLicense("<Put your license key here>");

//Change the runtime settings to read both normal and inverted barcodes
reader->GetRuntimeSettings(runtimeSettings);
runtimeSettings->furtherModes.grayscaleTransformationModes[0] = GTM_ORIGINAL;
runtimeSettings->furtherModes.grayscaleTransformationModes[1] = GTM_INVERTED;
reader->UpdateRuntimeSettings(runtimeSettings, sError, 512);

//Replace "<Put your file path here>" with your own file path
reader->DecodeFile("<Put your file path here>", "");
// If succeeds
reader->GetAllTextResults(&paryResult);
CBarcodeReader::FreeTextResults(&paryResult);
delete runtimeSettings;
delete reader;
```

Code snippet in java:

```java
BarcodeReader dbr = new BarcodeReader();
PublicRuntimeSettings rts = dbr.getRuntimeSettings();
//Change the runtime settings to read both normal and inverted barcodes
rts.furtherModes.grayscaleTransformationModes[0]= EnumGrayscaleTransformationMode.GTM_ORIGINAL;
rts.furtherModes.grayscaleTransformationModes[1]= EnumGrayscaleTransformationMode.GTM_INVERTED;
dbr.updateRuntimeSettings(rts);
//Initialize license prior to any decoding
dbr.initLicense("Put your license key here");
TextResult[] result = dbr.decodeFile("Put your file path here","");
```

How to filter out unwanted barcode results

Dynamsoft Barcode Reader SDK is able to read multiple barcodes at once and return results of all the decoded barcodes. However, you may not want all the results. For example, you may need only the results of a specific barcode format, or you may need only the barcodes with a certain length (number of barcode data bytes). The SDK provides settings to help you filter out the barcode results by barcode formats, confidence and text length.

Filtering out the barcode results based on the barcode format is shown in Specify Which Barcode Type to Read.

The following code shows how to filter out the unwanted barcode results based on the barcode confidence and barcode text length.

- Filtering using barcode confidence
- Filtering using barcode result length

Filtering using barcode confidence

The barcode confidence means the probablity that the barcode result is reconigzed correctly. You can use `minResultConfidence` to filter out the results with low confidence.

Code snippet in c:

```c
void *hBarcode = NULL;
char sError[512];
TextResultArray* pResult = NULL;
PublicRuntimeSettings runtimeSettings;
hBarcode = DBR_CreateInstance();
// Initialize license prior to any decoding
//Replace "<Put your license key here>" with your own license
DBR_InitLicense(hBarcode, "<Put your license key here>");

// Obtain the barcode results with confidence above 35
DBR_GetRuntimeSettings(hBarcode, &runtimeSettings);
runtimeSettings.minResultConfidence = 35;  //It is recommended to set the confidence above 35
DBR_UpdateRuntimeSettings(hBarcode, &runtimeSettings,sError,512);

// Replace "<Put the path of your file here>" with your own file path
DBR_DecodeFile(hBarcode,"<Put your file path here>","");
DBR_GetAllTextResults(hBarcode, &pResult);
DBR_FreeTextResults(&pResult);
DBR_DestroyInstance(hBarcode);
```

Code snippet in cpp:

```cpp
char sError[512];
TextResultArray* paryResult = NULL;
PublicRuntimeSettings* runtimeSettings = new PublicRuntimeSettings();
CBarcodeReader* reader = new CBarcodeReader();
// Initialize license prior to any decoding
// Replace "<Put your license key here>" with your own license
reader->InitLicense("<Put your license key here>");

// Obtain the barcode results with confidence above 35
reader->GetRuntimeSettings(runtimeSettings);
runtimeSettings->minResultConfidence = 35;    //It is recommended to set the confidence above 35
reader->UpdateRuntimeSettings(runtimeSettings, sError, 512);
```

```
// Replace "<Put your file path here>" with your own file path
reader->DecodeFile("<Put your file path here>", "");
// If succeeds
reader->GetAllTextResults(&paryResult);
CBarcodeReader::FreeTextResults(&paryResult);
delete runtimeSettings;
delete reader;
```

Code snippet in java:

```
BarcodeReader dbr = new BarcodeReader();
PublicRuntimeSettings rts = dbr.getRuntimeSettings();

rts.minResultConfidence = 35;//It is recommended to set the confidence above 35
dbr.updateRuntimeSettings(rts);

// Initialize license prior to any decoding
//Replace "<Put your license key here>" with your own license
dbr.initLicense("<Put your license key here>");

//Replace "<Put the path of your file here>" with your own file path
TextResult[] result = dbr.decodeFile("<Put your file path here>","");
```

Filtering using barcode result length

The barcode length is calculated in bytes. The length of the barcode text does not necessarily mean the actual length of the barcode bytes. You can use `minBarcodeTextLength` to filter out some invalid or unwanted results.

Code snippet in c:

```
void *hBarcode = NULL;
char sError[512];
TextResultArray* pResult = NULL;
PublicRuntimeSettings runtimeSettings;
hBarcode = DBR_CreateInstance();
// Initialize license prior to any decoding
// Replace "<Put your license key here>" with your own license
DBR_InitLicense(hBarcode, "<Put your license key here>");

// Decode the barcodes with its length longer than 10 bytes
DBR_GetRuntimeSettings(hBarcode, &runtimeSettings);
runtimeSettings.minBarcodeTextLength = 10; //The length is calculated in bytes
DBR_UpdateRuntimeSettings(hBarcode, &runtimeSettings,sError,512);

// Replace "<Put your file path here>" with your own file path
DBR_DecodeFile(hBarcode,"put your file path here","");
DBR_GetAllTextResults(hBarcode, &pResult);
DBR_FreeTextResults(&pResult);
DBR_DestroyInstance(hBarcode);
```

Code snippet in cpp:

```
char sError[512];
TextResultArray* paryResult = NULL;
PublicRuntimeSettings* runtimeSettings = new PublicRuntimeSettings();
CBarcodeReader* reader = new CBarcodeReader();
// Initialize license prior to any decoding
// Replace "Put your license key here" with your own license
reader->InitLicense("Put your license key here");
```

```
// Decode the barcodes with its length longer than 10 bytes
reader->GetRuntimeSettings(runtimeSettings);
runtimeSettings->minBarcodeTextLength = 10;
reader->UpdateRuntimeSettings(runtimeSettings, sError, 512);

// Replace "Put the path of your file here" with your own file path
reader->DecodeFile("Put your file path here", "");
// If succeeds
reader->GetAllTextResults(&paryResult);
CBarcodeReader::FreeTextResults(&paryResult);
delete runtimeSettings;
delete reader;
```

Code snippet in java:

```
BarcodeReader dbr = new BarcodeReader();
PublicRuntimeSettings rts = dbr.getRuntimeSettings();

rts.minBarcodeTextLength = 10; //The length is calculated in bytes
dbr.updateRuntimeSettings(rts);

// Initialize license prior to any decoding
//Replace "<Put your license key here>" with your own license
dbr.initLicense("<Put your license key here>");

//Replace "<Put the path of your file here>" with your own file path
TextResult[] result = dbr.decodeFile("<Put your file path here>","");
```

How to decode DPM Data Matrix

Since version 7.2 of Dynamsoft Barcode Reader SDK, direct part mark (DPM) Data Matrix scanning is supported. To decode DPM codes, some particular settings are required:

1. The value `DPMCRM_GENERAL` needs to be set for the parameter `PublicRuntimeSettings->FurtherModes->DPMCodeReadingModes`.
2. The value `LM_STATISTICS_MARKS` needs to be set for the parameter `PublicRuntimeSettings->LocalizationModes`.

The following code shows how to set the runtime settings for DPM decoding:

Code snippet in c:

```c
void *hBarcode = NULL;
char sError[512];
TextResultArray* pResult = NULL;
hBarcode = DBR_CreateInstance();
DBR_InitLicense(hBarcode,"put your license here");
DBR_GetRuntimeSettings(hBarcode, &runtimeSettings);
//turn on the DPM mode
runtimeSettings.furtherModes.dpmCodeReadingModes[0] = DPMCRM_GENERAL;
runtimeSettings.localizationModes[0] = LM_STATISTICS_MARKS;
//update the runtime settings
DBR_UpdateRuntimeSettings(hBarcode, &runtimeSettings, sError, 512);
DBR_DecodeFile(hBarcode, "put your file path here", "");
DBR_GetAllTextResults(hBarcode, &pResult);
DBR_FreeTextResults(&pResult);
DBR_DestroyInstance(hBarcode);
```

Code snippet in cpp:

```cpp
char sError[512];
TextResultArray* paryResult = NULL;
CBarcodeReader reader;
PublicRuntimeSettings* runtimeSettings = new PublicRuntimeSettings();
reader.InitLicense("<your license key here>");
reader.GetRuntimeSettings(runtimeSettings);
//turn on the DPM mode
runtimeSettings->furtherModes.dpmCodeReadingModes[0] = DPMCRM_GENERAL;
runtimeSettings->localizationModes[0] = LM_STATISTICS_MARKS;
//update the runtime settings
reader.UpdateRuntimeSettings(runtimeSettings, sError, 512);
reader.DecodeFile("<your image file full path>", "");
reader.GetAllTextResults(&paryResult);
CBarcodeReader::FreeTextResults(&paryResult);
```

Code snippet in java:

```java
BarcodeReader dbr = new BarcodeReader("put your license here");
PublicRuntimeSettings settings = dbr.getRuntimeSettings();
settings.furtherModes.dpmCodeReadingModes[0] = EnumDPMCodeReadingMode.DPMCRM_GENERAL;
settings.localizationModes[0]= EnumLocalizationMode.LM_STATISTICS_MARKS;
dbr.updateRuntimeSettings(settings);
TextResult[] results = dbr.decodeFile("put your file path here", "");
```

How to get the angle of the barcodes

Dynamsoft Barcode Reader SDK is able to detect barcodes at all angles. The SDK is also able to return the angles of the barcodes decoded. The result is stored in the struct `LocalizationResult`. The following code snippet shows how to get the rotation/skew angles of the barcodes decoded by the SDK.

Code snippet in c:

```c
void *hBarcode = NULL;
int iIndex = 0;
TextResultArray* pResult = NULL;
hBarcode = DBR_CreateInstance();
DBR_InitLicense(hBarcode,"put your license here");
DBR_DecodeFile(hBarcode, "put your file path here", "");
DBR_GetAllTextResults(hBarcode, &pResult);
for (iIndex = 0; iIndex < pResult->resultsCount; iIndex++)
{
printf("Result %d\n", iIndex + 1);
printf("Barcode angle:$d \n", pResult->results[iIndex]->localizationResult->angle);
}
DBR_FreeTextResults(&pResult);
DBR_DestroyInstance(hBarcode);
```

Code snippet in cpp:

```cpp

TextResultArray* paryResult = NULL;
CBarcodeReader reader;
reader.InitLicense("put your license key here");
reader.DecodeFile("put your image file full path here", "");
reader.GetAllTextResults(&paryResult);
printf("%d total barcodes found. \r\n", paryResult->resultsCount);
for (int iIndex = 0; iIndex < paryResult->resultsCount; iIndex++)
{
printf("Result %d\r\n", iIndex + 1);
printf("Barcode Angle:%d \n", paryResult->results[iIndex]->localizationResult->angle);
}
CBarcodeReader::FreeTextResults(&paryResult);
```

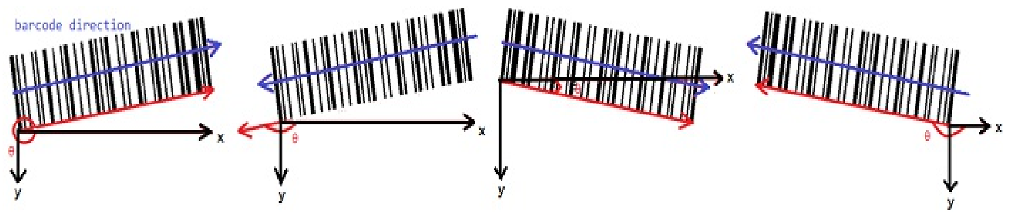
Code snippet in java:

```java
BarcodeReader dbr = new BarcodeReader("put your license here");
TextResult[] results = dbr.decodeFile("put your file path here", "");
String output = "";
for(int i =0; i<result.length;i++){
output += "Barcode Angle: ";
output += result[i].localizationResult.angle + "\n\n";
}
System.out.println(output);
```

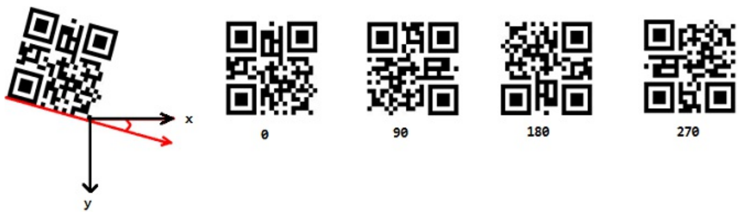
Angles for different barcodes

The following illustrations will show how the angle is calculated for different barcode types.

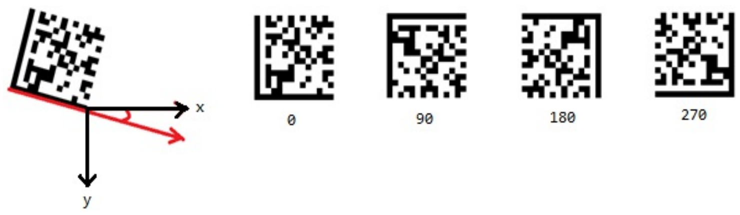
1. OneD barcode



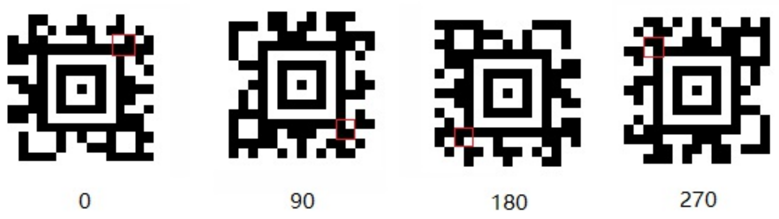
2. QR



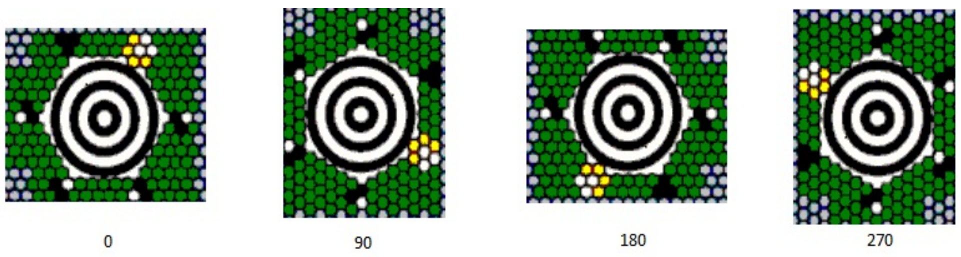
3. Data Matrix



4. Aztec



5. Maxicode



Get barcode confidence

The score of recognition confidence could measure the reliability of a recognized result. The higher the score, the more precise the results are. We could obtain confidence result from `ExtendedResult`. The following code snippet shows how to get the confidence of the barcodes decoded by the SDK.

Code snippet in c:

```
void *hBarcode = NULL;
int iIndex = 0;
TextResultArray* pResult = NULL;
hBarcode = DBR_CreateInstance();
DBR_InitLicense(hBarcode,"put your license here");
DBR_DecodeFile(hBarcode, "put your file path here", "");
DBR_GetAllTextResults(hBarcode, &pResult);
for (iIndex = 0; iIndex < pResult->resultsCount; iIndex++)
{
printf("Result %d\n", iIndex + 1);
printf("Barcode Confidence : %d \r", pResult->results[iIndex]->results[0]->confidence);
DBR_FreeTextResults(&pResult);
DBR_DestroyInstance(hBarcode);
```

Code snippet in cpp:

```

TextResultArray* paryResult = NULL;
CBarcodeReader reader;
reader.InitLicense("put your license key here");
reader.DecodeFile("put your image file full path here", "");
reader.GetAllTextResults(&paryResult);
printf("%d total barcodes found. \r\n", paryResult->resultsCount);
for (int iIndex = 0; iIndex < paryResult->resultsCount; iIndex++)
{
printf("Result %d\r\n", iIndex + 1);
printf("Barcode Confidence : %d \r", paryResult->results[iIndex]->results[0]->confidence);
CBarcodeReader::FreeTextResults(&paryResult);
```

Code snippet in csharp:

```
BarcodeReader dbr = new BarcodeReader();
PublicRuntimeSettings runtimeSettings = dbr.GetRuntimeSettings();
dbr.ProductKeys = "Put your license key here";
TextResult[] aryResult = dbr.DecodeFile("Put your file path here", "");
for (var i = 0; i < aryResult.Length; i++)
{
Console.WriteLine("Barcode: {0}", (i + 1));
Console.WriteLine("Barcode Confidence: {0}", aryResult[i].Results[0].confidence);
}
```

Code snippet in java:

```
BarcodeReader dbr = new BarcodeReader("put your license here");
TextResult[] results = dbr.decodeFile("put your file path here", "");
String output = "";
for(int i =0; i<result.length;i++){
output += "Barcode Confidence: ";
output += result[i].results[0].confidence + "\n\n";
}
```

```
System.out.println(output);
```

How to get intermediate results

Apart from getting the results like barcode type, value, location, Dynamsoft Barcode Reader also provides APIs for you to get the intermediate results like original image, transformed grayscale image, binarized image, text zone, etc. for further analysis. To learn more about what intermediate results you can get, see enum IntermediateResultType.

Note: You will need a separate license for the intermediate results such as transformed grayscale image, binarized image, and text zone. Getting the original image of the intermediate result doesn't require a separate license. The original image is the one that Dynamsoft Barcode Reader SDK uses to do the barcode reading. If the SDK fails to recognize the barcodes using the original image, try different settings or contact Dynamsoft Support for help.

Here we will show how to get the original image and save it to your file system.

```
    int iRet = -1;
    hBarcode = DBR_CreateInstance();
    DBR_InitLicense(hBarcode, "t0068MgAAAGotN***"); // Set the trial license

    DBR_GetRuntimeSettings(hBarcode,&runtimeSettings);
    runtimeSettings.intermediateResultTypes = IRT_ORIGINAL_IMAGE;
    // Here we save it to the file system. You can also save it to memory according to your needs.
    runtimeSettings.intermediateResultSavingMode = IRSM_FILESYSTEM;
    iRet = DBR_UpdateRuntimeSettings(hBarcode,&runtimeSettings,szErrorMsg,256);
    // Set the folder path which stores the intermediate result (original image). Please make sure you have write
 permission to this folder.
    iRet = DBR_SetModeArgument(hBarcode, "IntermediateResultSavingMode", 0, "FolderPath", "D:\DBRLogs", szErrorMs
g, 256);
    if (iRet != DBR_OK) // If error occurs
    {
        printf("Error code: %d. Error message: %s\n", iRet, szErrorMsg);
        return -1;
    }
    DBR_DecodeFile(hBarcode, "<Your Image Path here>","");
    IntermediateResultArray* pResults;
    DBR_GetIntermediateResults(hBarcode, &pResults);

    DBR_DestroyInstance(hBarcode);
```

Get detailed barcode information

The Dynamsoft Barcode Reader SDK provides APIs for you to get the detailed barcode information like checksum digit, start/stop characters, error correction level, etc. To learn more about what information you can get, see the following API links:

- Class OneDCodeDetails
- Class QRCodeDetails
- Class PDF417Details
- Class DataMatrixDetails
- Class AztecDetails

Here we take QR Code as example and show how to get the version and model of a QR Code.

What is the version of a QR Code?

QRCode Version	Modules
Version 1	21 x 21
Version 2	25 x 25
...	...
Version N	(17 + N x 4) x (17 + N x 4)
Version 40	177 x 177

What is the model of a QR Code?

QRCode Model	Description
Model 1	The original QR Code. It is a code capable of coding 1,167 numerals with its maximum version being 14 (73 x 73 modules).
Model 2	Created by improving Model 1 so that this code can be read smoothly even if it is distorted in some way. This code can encode up to 7,089 numerals with its maximum version being 40 (177 x 177 modules). Today, the term QRCode usually refers to QRCode Model 2.

Code snippet

Code snippet in csharp:

```
BarcodeReader dbr = new BarcodeReader();
dbr.ProductKeys = "t0068MgAAAGotN***"; // Set the trial license
PublicRuntimeSettings settings = dbr.GetRuntimeSettings();
settings.BarcodeFormatIds = (int)EnumBarcodeFormat.BF_QR_CODE; // Read QR Code only
dbr.UpdateRuntimeSettings(settings);
// leave the template name empty ("") will use default settings for recognition
TextResult[] aryResult = dbr.DecodeFile(@"C:\Program Files (x86)\Dynamsoft\Barcode Reader 7.2\Images\AllSupported
BarcodeTypes.tif", "");
if (aryResult.Length == 0)
{
    Console.WriteLine("No barcode found.");
}
else
{
    for (var i = 0; i < aryResult.Length; i++)
    {
        QRCodeDetails details = (QRCodeDetails)aryResult[i].DetailedResult;
        Console.WriteLine("Barcode Index: {0}", (i + 1));
        Console.WriteLine("BarcodeFormat: {0}", aryResult[i].BarcodeFormat.ToString());
        Console.WriteLine("BarcodeText: {0}", aryResult[i].BarcodeText);
        Console.WriteLine("QRCode Model: {0}", details.Model.ToString()); // Get the Model
        Console.WriteLine("QRCode Version: {0}", details.Version.ToString()); // Get the Version
    }
}
```

Code snippet in c:

```
// Define variables
void* hBarcode = NULL;
int iRet = -1;
int iIndex = 0;
int iLicMsg = -1;
char sError[512];
TextResultArray* pResult = NULL;
hBarcode = DBR_CreateInstance();
PublicRuntimeSettings runtimeSettings;
DBR_GetRuntimeSettings(hBarcode, &runtimeSettings);
runtimeSettings.barcodeFormatIds = BF_QR_CODE; // OneD barcode
DBR_UpdateRuntimeSettings(hBarcode, &runtimeSettings, sError, 512);
// Initialize license prior to any decoding
iLicMsg = DBR_InitLicense(hBarcode, "f0068MgAAAGnTM***");
//If error occurs to the license initialization
if (iLicMsg != DBR_OK)
{
    printf("Failed to initialize the license successfully: %d\r\n%s\r\n", iLicMsg, DBR_GetErrorString(iLicMsg));
    return iLicMsg;
}
// Started decoding
DBR_DecodeFile(hBarcode, "<input your image file path here>", "");
// Get the text result
iRet = DBR_GetAllTextResults(hBarcode, &pResult);
// If error occurs
if (iRet != DBR_OK)
{
    printf("Failed to read barcode: %d\r\n%s\r\n", iRet, DBR_GetErrorString(iRet));
    return iRet;
}
// If succeeds
printf("%d total barcode(s) found. \n", pResult->resultsCount);
for (iIndex = 0; iIndex < pResult->resultsCount; iIndex++)
{
    QRCodeDetails* qrd = (QRCodeDetails*)pResult->results[iIndex]->detailedResult;
    printf("QRCode Model:%d \r\n", qrd->model);
    printf("QRCode Version: %d \r\n", qrd->version);
    printf("Result %d\n", iIndex + 1);
    printf("Barcode Format: %s\n", pResult->results[iIndex]->barcodeFormatString);
    printf("Text reads: %s \n", pResult->results[iIndex]->barcodeText);
}
// Finally release BarcodeResultArray and destroy instance
DBR_FreeTextResults(&pResult);
DBR_DestroyInstance(hBarcode);
system("pause");
```

Code snippet in cpp:

```
int iRet = -1;
int iLicMsg = -1;
TextResultArray* paryResult = NULL;
PublicRuntimeSettings* runtimeSettings = new PublicRuntimeSettings();
```

```
// Initialize license prior to any decoding
CBarcodeReader reader;
char sError[512];
iLicMsg = reader.InitLicense("f0068MgA4o***");
reader.GetRuntimeSettings(runtimeSettings);
runtimeSettings->barcodeFormatIds = BF_QR_CODE;
reader.UpdateRuntimeSettings(runtimeSettings, sError, 512);
//If error occurs to the license initialization
if (iLicMsg != DBR_OK)
{
    printf("Failed to initialize the license successfully: %d\r\n%s\r\n", iLicMsg, DBR_GetErrorString(iLicMsg));
    return iLicMsg;
}
// Start decoding
iRet = reader.DecodeFile("<please input your image file path>", "");
// If error occurs
if (iRet != DBR_OK)
{
    printf("Failed to read barcode: %d\r\n%s\r\n", iRet, DBR_GetErrorString(iRet));
    return iRet;
}
// If succeeds
reader.GetAllTextResults(&paryResult);
for (int iIndex = 0; iIndex < paryResult->resultsCount; iIndex++)
{
    QRCodeDetails* qrd = (QRCodeDetails*)paryResult->results[iIndex]->detailedResult;
    printf("QRCode Model:%d \r\n", qrd->model);
    printf("QRCode Version: %d \r\n", qrd->version);
    printf("BarcodeFormat: %s\r\n", paryResult->results[iIndex]->barcodeFormatString);
    printf("Text read: %s\r\n", paryResult->results[iIndex]->barcodeText);
}
// Finally release BarcodeResultArray
CBarcodeReader::FreeTextResults(&paryResult);
return 0;
```

How to turn on or off text filter

In some cases, the image may be filled with a lot of characters and numbers. This could affect the barcode recognition. With text filter on, it can filter and remove the characters and numbers from the image and improve the barcode recognition accuracy. However, you may want to turn off text filter when barcodes are the only content of the image or when the speed is the prority.

The text filter is on by default.

The following code snippet shows how to set text filter function by changing the `TextFilterModes` parameter.

Code snippet in c:

```
void *hBarcode = NULL;
char sError[512];
TextResultArray* pResult = NULL;
PublicRuntimeSettings runtimeSettings;
hBarcode = DBR_CreateInstance();
// Initialize license prior to any decoding
//Replace "<Put your license key here>" with your own license
DBR_InitLicense(hBarcode, "<Put your license key here>");

DBR_GetRuntimeSettings(hBarcode, &runtimeSettings);
//Turn off the text filter function. The enumeration includes TFM_GENERAL_CONTOUR,
//which is used to turn on the function.
runtimeSettings.furtherModes.textFilterModes[0] = TFM_SKIP;
DBR_UpdateRuntimeSettings(hBarcode, &runtimeSettings,sError,512);

//Replace "<Put your file path here>" with your own file path
DBR_DecodeFile(hBarcode,"put your file path here","");
DBR_GetAllTextResults(hBarcode, &pResult);
DBR_FreeTextResults(&pResult);
DBR_DestroyInstance(hBarcode);
```

Code snippet in cpp:

```
char sError[512];
TextResultArray* paryResult = NULL;
PublicRuntimeSettings* runtimeSettings = new PublicRuntimeSettings();
CBarcodeReader* reader = new CBarcodeReader();
//Initialize license prior to any decoding
//Replace "Put your license key here" with your own license
reader->InitLicense("Put your license key here");

reader->GetRuntimeSettings(runtimeSettings);
//Turn off the text filter function. The enumeration includes TFM_GENERAL_CONTOUR,
//which is used to turn on the function, and TFM_AUTO.
runtimeSettings->furtherModes.textFilterModes[0] = TFM_SKIP;
reader->UpdateRuntimeSettings(runtimeSettings, sError, 512);

//Replace "Put the path of your file here" with your own file path
reader->DecodeFile("Put your file path here", "");
// If succeeds
reader->GetAllTextResults(&paryResult);
CBarcodeReader::FreeTextResults(&paryResult);
delete runtimeSettings;
delete reader;
```

Code snippet in java:

```java
BarcodeReader dbr = new BarcodeReader();
PublicRuntimeSettings rts = dbr.getRuntimeSettings();

rts.furtherModes.textFilterModes[0] = EnumTextFilterMode.TFM_SKIP;
dbr.updateRuntimeSettings(rts);

// Initialize license prior to any decoding
//Replace "<Put your license key here>" with your own license
dbr.initLicense("<Put your license key here>");

//Replace "<Put the path of your file here>" with your own file path
TextResult[] result = dbr.decodeFile("<Put your file path here>","");
```

How to scan in multiple threads

To scan barcodes in multiple threads using Dynamsoft Barcode Reader, you need to create multiple instances of `BarcodeReader` and run separate instance in each thread. Dynamsoft Barcode Reader SDK is non-thread safe. Please don't have multiple threads access the same `BarcodeReader` object.

We have a sample that shows how to use multiple threads to read barcodes from images using the C API.

Get the sample: Multi-thread scanning in C >

Note: The sample is for Windows Edition, but the C code is the same as the Linux Edition. You can refer to the source code for reference and build your own application.

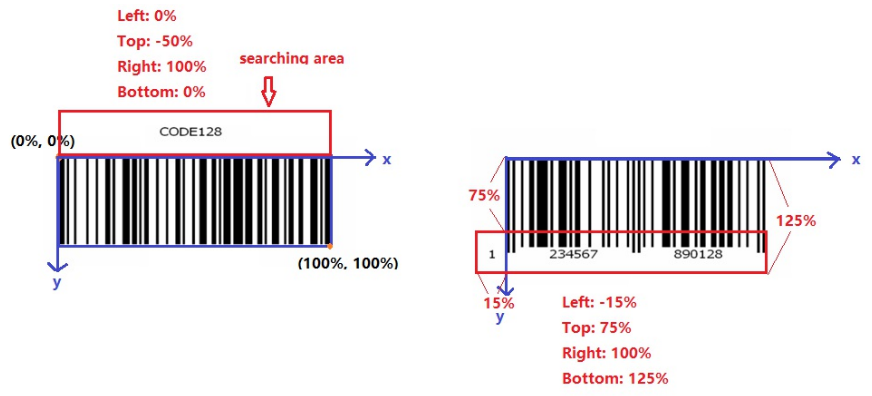
How to set custom area for accompanying texts

In some circumstances (especially for 1D barcodes), a barcode is typically accompanied by a human readable text. Since version 7.3, Dynamsoft Barcode Reader SDK is able to recognize the accompanying human readable text of a barcode. This text is useful for users to check whether the returned barcode result is correct.

By default, our library will check the top and bottom areas of the barcode for accompanying texts. However, you can set a custom area via the following parameters:

Mode Parameter Name	Argument Name	Argument Description	Value Range	Default Value
AccompanyingTextRecognitionModes	RegionBottom	Specifies the y-coordinate of the bottom-right corner of the region in percentage. This value is relative to the top-left corner of the barcode.	[-255, 255]	0
AccompanyingTextRecognitionModes	RegionLeft	Specifies the x-coordinate of the top-left corner of the region in percentage. This value is relative to the top-left corner of the barcode.	[-255, 255]	0
AccompanyingTextRecognitionModes	RegionRight	Specifies the x-coordinate of the bottom-right corner of the region in percentage. This value is relative to the top-left corner of the barcode.	[-255, 255]	0
AccompanyingTextRecognitionModes	RegionTop	Specifies the y-coordinate of the top-left corner of the region in percentage. This value is relative to the top-left corner of the barcode.	[-255, 255]	0

To specify the region of the accompanying texts, you need to set `PublicRuntimeSettings->FurtherModes->AccompanyingTextRecognitionModes` to `ATRM_GENERAL` and then specify `AccompanyingTextRecognitionModes.RegionLeft` , `AccompanyingTextRecognitionModes.RegionLeft` , `AccompanyingTextRecognitionModes.RegionLeft` and `AccompanyingTextRecognitionModes.RegionLeft` individually. The following diagram shows examples on how to set the value:



Below is the code snippet on how to set custom area for accompanying texts in Java:

```
BarcodeReader br = new BarcodeReader();

PublicRuntimeSettings runtimeSettings = br.getRuntimeSettings();
runtimeSettings.expectedBarcodesCount = iMaxCount;
runtimeSettings.barcodeFormatIds = barcodeFormatIds;
runtimeSettings.barcodeFormatIds_2 = barcodeFormatIds_2;
runtimeSettings.furtherModes.accompanyingTextRecognitionModes[0] = EnumAccompanyingTextRecognitionMode.ATRM_GENER
AL;
br.updateRuntimeSettings(runtimeSettings);
br.setModeArgument("AccompanyingTextRecognitionModes", 0, "RegionBottom", "150");
br.setModeArgument("AccompanyingTextRecognitionModes", 0, "RegionLeft", "-10");
br.setModeArgument("AccompanyingTextRecognitionModes", 0, "RegionRight", "85");
br.setModeArgument("AccompanyingTextRecognitionModes", 0, "RegionTop", "100");

TextResult[] results = br.decodeFile("<put your image file here>", "");
String pszTemp = String.format("Total barcode(s) found: %d. Total time spent: %.3f seconds.", results.length, ((f
loat) (ullTimeEnd - ullTimeBegin) / 1000));
iIndex = 0;
for (TextResult result : results) {
    iIndex++;
    pszTemp = String.format("  Barcode %d:", iIndex);
    System.out.println(pszTemp);
    pszTemp = "    Value: " + result.barcodeText;
    System.out.println(pszTemp);
    byte[] byteArray = result.results[0].accompanyingTextBytes;
    String str = new String(byteArray);
    System.out.println("accompanyingTextBytes : " + str);
}
```

How to enable scale up for barcode recognition

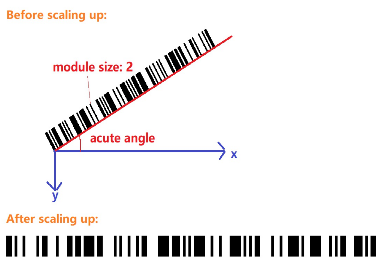
For some barcodes with small module size, our library will automatically enlarge the barcode to a proper size before recognition.

Since version 7.3, Dynamsoft Barcode Reader SDK provides `PublicRuntimeSettings->ScaleUpModes` , `ModuleSizeThreshold` , `AcuteAngleWithXThreshold` and `TargetModuleSize` APIs which allow you to set the custom rules for scaling up. To enable it, you need to set `PublicRuntimeSettings->ScaleUpModes` to `SUM_LINEAR_INTERPOLATION` or `SUM_NEAREST_NEIGHBOUR_INTERPOLATION` , then set the rest parameters.

If the module size of the barcode < `ModuleSizeThreshold` and the acute angle with X of the barcode > `AcuteAngleWithXThreshold` , the barcode will be enlarged to N times (N=1,2,3...) till N * modulesize >= `TargetModuleSize` .

Example

Condition: `ModuleSizeThreshold` = 4, `AcuteAngleWithXThreshold` = 30, `TargetModuleSize` = 4



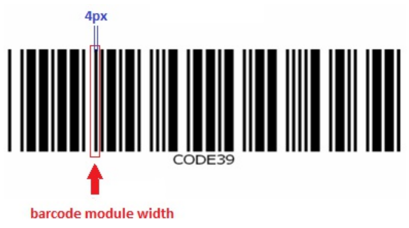
Result:

The module size of the barcode in the image is 2, which is smaller than `ModuleSizeThreshold` , and the acute angle is larger than `AcuteAngleWithXThreshold` , so our library will perform scaling up operation. After scaling up, the barcode is enlarged to 2 times since 2 * modulesize >= `TargetModuleSize` .

Check if the barcode image is clear enough for recognition?

We often receive questions from customers as to why Dynamsoft Barcode Reader SDK is unable to decode their barcode, and the answer is often that the resolution of the original image is too low. Then what is the right resolution for a barcode image?

The resolution alone does not decide the barcode readability but what does matter is how wide the barcode module (x dimension of barcode module width of the tiniest bar) is in the image, as shown in the image below:

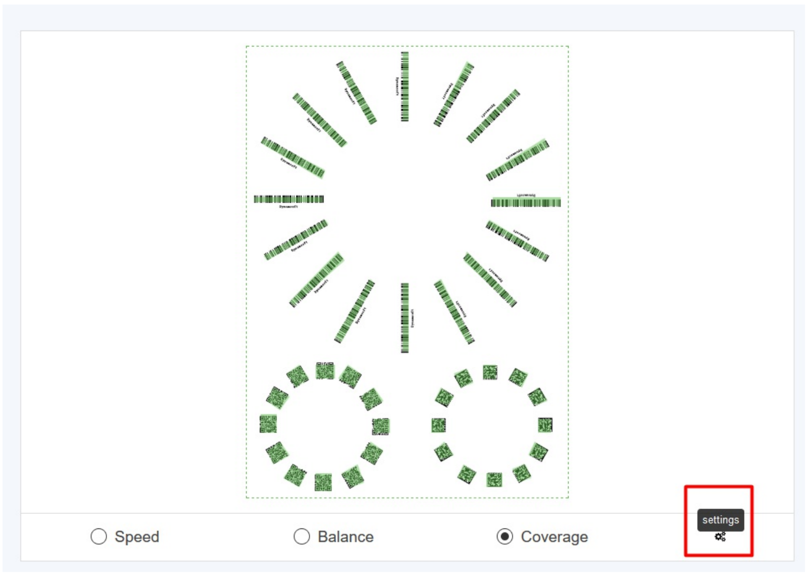


The recommended barcode module width for each barcode type is at least **2 pixels**.

Generate a custom barcode reading template

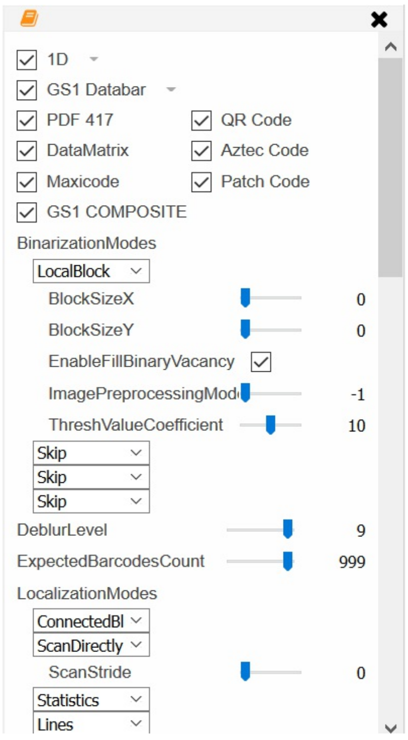
Starting from Dynamsoft Barcode Reader version 6, it is recommended to set all barcode reading configurations in the template (*.json). This article will talk about how to customize your own template based on our standard templates on our online demo.

1. Open our Online Demo.



Speed [✓ Coverage] Balance

2. Modify the settings on the right panel.



- [✓] 1D
- [✓] GS1 Databar
- [✓] PDF 417 — [✓] QR Code
- [✓] DataMatrix — [✓] Aztec Code
- [✓] Maxicode — [✓] Patch Code
- [✓] GS1 COMPOSITE

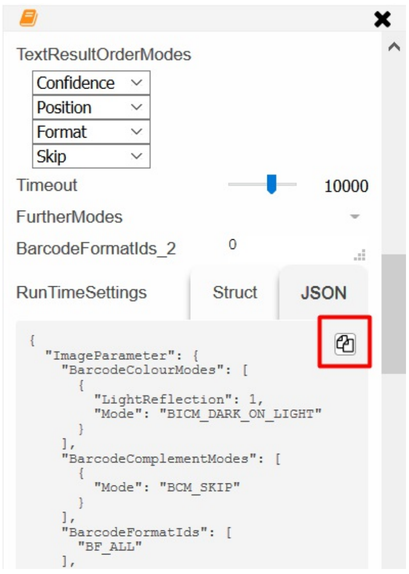
BinarizationModes
- LocalBlock
  - BlockSizeX 0
  - BlockSizeY 0
  - EnableFillBinaryVacancy [✓]
  - ImagePreprocessingModeIndex -1
  - ThreshValueCoefficient 10
- Skip
- Skip
- Skip

DeblurLevel 9

ExpectedBarcodesCount 999

LocalizationModes
- ConnectedBl
- ScanDirectly
  - ScanStride 0
- Statistics
- Lines

3. At the bottom of the panel, switch to **JSON** tab and click the **copy** button.


TextResultOrderModes

Confidence
Position
Format
Skip

Timeout 10000

FurtherModes

BarcodeFormatIds_2 0

RunTimeSettings

```
{
  "ImageParameter": {
    "BarcodeColourModes": [
      {
        "LightReflection": 1,
        "Mode": "BICM_DARK_ON_LIGHT"
      }
    ],
    "BarcodeComplementModes": [
      {
        "Mode": "BCM_SKIP"
      }
    ],
    "BarcodeFormatIds": [
      "BF_ALL"
    ],
```

Save the string as a .json file and use `InitRuntimeSettingsWithFile()` method to update the barcode reading settings. Please refer to Use a template to change settings for code snippets.

Licensing and Distributing

This section covers the following topics:

Use a trial key

To use a trial license, you can follow the steps below:

1. Get a trial license key.

   If you installed Dynamsoft Barcode Reader 30-day free trial, it comes with a 30-day trial license by default.

   For Windows Edition, you can find the license key in the License Manager program at */{INSTALLATION FOLDER}/LicenseManager.exe*.

   Note: If the trial license expires or it is missing, you can still get barcode reading results but partial of the result will be masked with "".* You may log in the customer portal and request for a trial extension online.

2. Update the license key in source code.

   You can use `initLicense()` or `ProductKeys` to set the license.

Code snippet in c:

```c
void *hBarcode = NULL;
hBarcode = DBR_CreateInstance();
DBR_InitLicense(hBarcode, "t0068NQAAAI8+mMcYRNwmijAzExhq******");
DBR_DestroyInstance(hBarcode);
```

Code snippet in cpp:

```cpp
CBarcodeReader reader = new CBarcodeReader();
reader.InitLicense("t0068NQAAAI8+mMcYRNwmijAzExhq******");
```

Code snippet in csharp:

```csharp
BarcodeReader reader = new BarcodeReader();
reader.ProductKeys = "t0068NQAAAI8+mMcYRNwmijAzExhq******";
```

Code snippet in vbnet:

```vbnet
Dim reader As BarcodeReader = New Dynamsoft.Barcode.BarcodeReader()
reader.ProductKeys = "t0068NQAAAI8+mMcYRNwmijAzExhq******"
```

Code snippet in java:

```java
BarcodeReader mBarcodeReader;
mBarcodeReader = new BarcodeReader("t0068NQAAAI8+mMcYRNwmijAzExhq******");
```

Code snippet in php:

```php
$br = new BarcodeReader();
$br->initLicense("t0068NQAAAI8+mMcYRNwmijAzExhq******");
```

Code snippet in py:

```python
def initLicense(license):
    dbr.initLicense(license)

initLicense("t0068NQAAAI8+mMcYRNwmijAzExhq******")
```

1. Save and rebuild your application.

Use a Development License

> The following guide is for C/C++/C#/Java/Python. If you are using PHP, please contact Dynamsoft for more info on how to set the license.

Different methods are used for setting trial and full license keys. In the demo or sample applications, we use `.InitLicense()` or `.ProductKeys` to set trial license keys. For the purchased version, you need to use `initLicenseFromServer()` or `initLicenseFromLicenseContent()` to complete the license registration.

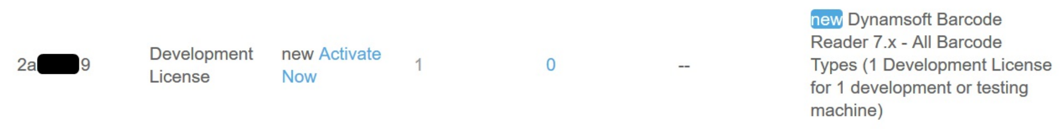
You can use a development license by following the steps below:

1. Activate a development license
2. Register the development license key

1. Activate a development license

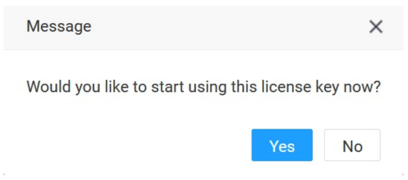
Once you obtain a Development License, you can find your license information at **Customer Portal** > **License Center** > **Barcode Reader SDK**.

To activate a development license (8-digit key), click the **Activate Now** link beside it.



2a[redacted]9	Development License	new Activate Now	1	0	--	new Dynamsoft Barcode Reader 7.x - All Barcode Types (1 Development License for 1 development or testing machine)

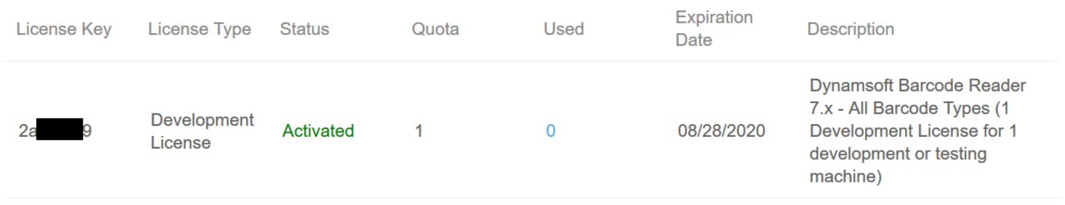
On the following popup window, click the **Yes** button.



Message

Would you like to start using this license key now?

Then you can see the status, quota, used seats and expiration date of the activated license key.



License Key	License Type	Status	Quota	Used	Expiration Date	Description
2a[redacted]9	Development License	Activated	1	0	08/28/2020	Dynamsoft Barcode Reader 7.x - All Barcode Types (1 Development License for 1 development or testing machine)

You can repeat the above steps to activate other license keys.

2. Register the development license key

Here are the possible options to register a development license:

- Option1: Connect to Dynamsoft server once and use the SDK offline
- Option2: Always connect to Dynamsoft server for license verification
- Option3: No Internet connection

Option1: Connect to Dynamsoft server once and use the SDK offline

If you wish to use the SDK offline after activating, please follow the steps below:

1. Use `initLicenseFromServer()` to connect to Dynamsoft Hosted server (or your own server) to obtain the license information. The machine needs Internet connection to complete the device registration for the first time.

2. Use `outputLicenseToString()` to get the information of the license and store the information to the development machine.
3. Use `initLicenseFromLicenseContent()` API to register the license.

Code snippet in c:

```
void* _br = NULL;
int iLicMsg = -1;
char info[1024];
_br = DBR_CreateInstance();
FILE *file;
// Check if there is a license file on the local machine. If yes, use the local license file; Otherwise, connect
to Dynamsoft Hosted server to verify the license.
if ((file = fopen("license.txt", "r")) == NULL)
{
  // Connect to the Dynamsoft server to verify the license
  iLicMsg = DBR_InitLicenseFromServer(_br, "", "licenseKey1;licenseKey2");
  // The second parameter is the IP of the license server. Leaving it  empty ("") means it will connect to Dynams
oft License Server for online  verification automatically.
  // If error occurs to the license
  if (iLicMsg != DBR_OK) {
     printf("Failed to initialize the license: %d\r\n%s\r\n", iLicMsg, DBR_GetErrorString(iLicMsg));
     return iLicMsg;
  }
  DBR_OutputLicenseToString(_br, info, 1024); // For the third parameter, the recommended length is 512 for one l
icense key.
  // If you have N license keys, please set it to N * 512.

  // If you wish to use SDK offline, store the license information in txt or other format
  FILE *fp = fopen("license.txt", "w");
  if (fp == 0){
     printf("can't open file\n");
     return 0;
  }
  fwrite(info, sizeof(char) * 1024, 1, fp);
  fclose(fp);
}
else{
  // Use the local license file and use Dynamsoft Barcode Reader SDK offline
  FILE *fp = fopen("license.txt", "r");
  fscanf(fp, "%s", &info);
  fclose(fp);
  iLicenMsg = DBR_InitLicenseFromLicenseContent(_br, "licenseKey1;licenseKey2",  info);
  // If error occurs to the license
  if (iLicMsg != DBR_OK) {
     printf("Failed to initialize the license successfully: %d\r\n%s\r\n", iLicMsg, DBR_GetErrorString(iLicMsg));
     return iLicMsg;
  }
}

// Barcode decoding happens here
//....
DBR_DestroyInstance(_br);
```

Code snippet in cpp:

```
CBarcodeReader* reader = new CBarcodeReader();
int iLicMsg = -1;
char info[1024];
string filePath= "license.txt";
// To be able to use the license key offline, you need to store the license file obtained from Dynamsoft server o
nce you use the API, InitLicenseFromServer.
fstream licenseFile;
```

```
licenseFile.open(filePath, ios::in);
// Check if there is a license file on the local machine. If not, connect to Dynamsoft Hosted server to verify th
e license. Otherwise, use the local license file.
if (!licenseFile)
{
   // Connect to Dynamsoft server to verify the license.
   iLicMsg = reader->InitLicenseFromServer("", "licenseKey1;licenseKey2");
   // The first parameter is the IP of the license server. Leaving it empty ("") means it will connect to Dynamso
ft License Server for online verification automatically.

   // If error occurs to the license
   if (iLicMsg != DBR_OK)
   {
      printf("Failed to initialize the license: %d\r\n%s\r\n", iLicMsg, CBarcodeReader::GetErrorString(iLicMsg));
      return iLicMsg;
   }
   // If you wish to use SDK offline, store the license information in TXT or other format
   reader->OutputLicenseToString(info, 1024); // For the third parameter, the recommended length is 512 for one l
icense key.
  // If you have N license keys, please set it to N * 512.

   ofstream licFileOut(filePath);
   licFileOut << info;
   licFileOut.close();
}
else
{
   // Use the local license file and use Dynamsoft Barcode Reader SDK offline
   ifstream licFileIn(filePath);
   licFileIn >> info;
   licFileIn.close();
   iLicMsg = reader->InitLicenseFromLicenseContent("licenseKey1;licenseKey2", info);
   // If error occurs to the license
   if (iLicMsg != DBR_OK)
   {
      printf("Failed to initialize the license successfully: %d\r\n%s\r\n", iLicMsg, CBarcodeReader::GetErrorStri
ng(iLicMsg));
      return iLicMsg;
   }
}
// Decode barcodes happens here
//....
delete reader;
```

Code snippet in csharp:

```
int iLicMsg = -1;
string path = @"Put your file path here";
// To be able to use the license key offline, you need to store the license file obtained from Dynamsoft server o
nce you use the API, InitLicenseFromServer.
BarcodeReader _br = new BarcodeReader();
// Check if there is a license file in the local machine. If not, connect to Dynamsoft Hosted server to verify th
e license. Otherwise, use the local license file.
if (!File.Exists(path))
{
   // Connect to Dynamsoft server to verify the license
   iLicMsg = _br.InitLicenseFromServer("", "licenseKey1;licenseKey2");
   // The first parameter is the string of the license server. Leaving it empty ("") means it will connect to Dyn
amsoft License Server for online verification.
   if(iLicMsg != 0)
   {
      Console.WriteLine("License error Code:",iLicMsg);
```

```
        return;
    }
    // If you wish to use SDK offline, store the license information as txt format
    string license = _br.OutputLicenseToString();
    File.WriteAllText(path, license);
}
else{
    // Use the local license file and use Dynamsoft Barcode Reader SDK
    string license = File.ReadAllText(path);
    iLicMsg = _br.InitLicenseFromLicenseContent("licenseKey1;licenseKey2",license);
    if(iLicMsg != 0)
    {
        Console.WriteLine("Error Code:",iLicMsg);
        return;
    }
}
// Decode barcodes happens here
//....
```

Code snippet in java:

```
try {
    File file = new File("`<Please insert your intended license file path here for licensing procedure`>");
    BarcodeReader reader = new BarcodeReader();
    // Check if there is a license file on the local machine. If not, connect to Dynamsoft Hosted server to verify
 the license. Otherwise, use the license file.
    if (!file.exists()){
        // Connect to Dynamsoft server to verify the license.
        reader.initLicenseFromServer("", "licenseKey1;licenseKey2");
        //The first parameter is the string of the license server. Leaving it empty ("") means it will connect to D
ynamsoft License Server for online verification.
        //The second parameter refer to 8-bit short key. You may buy our product with more than 2 supported barcode
 format types and in that case you need to list every license key divided by colon(;)
        //If you bought our product with single supported barcode format type, please just fill in your single lice
nse key in second parameter.
        // If you wish to use SDK offline, store the license information as txt format or in other format
        String license = reader.outputLicenseToString();
        PrintWriter pw = new PrintWriter(file);
        pw.print(license);
        pw.close();
    }
    else{
        // Use the local license file and use Dynamsoft Barcode Reader SDK
        byte[] encoded = Files.readAllBytes(file.toPath());
        String license = new String(encoded, "utf-8");
        reader.initLicenseFromLicenseContent("licenseKey1;licenseKey2",license);
    }
}catch(Exception e) { //if your license is invalid, a BarcodeReaderException will be throw out
System.out.println(e);
}
```

Code snippet in py:

```
def initLicenseFromServer(pLicenseServer,pLicenseKey):
    dbr.initLicenseFromServer(pLicenseServer,pLicenseKey)
def initLicenseFromLicenseContent(pLicenseKey,pLicenseContent):
    dbr.initLicenseFromLicenseContent(pLicenseKey,pLicenseContent)
def outputLicenseToString():
    content = dbr.outputLicenseToString()
    return content
```

```
#Check if there is a license file on the local machine. If not, connect to Dynamsoft Hosted server to verify the
license. Otherwise, use the local license file.
if os.path.exists(license.txt):
   #Use the local license file to activate the SDK
   with open("license.txt","r") as f:
      pLicenseContent = f.read()
   initLicenseFromLicenseContent("licenseKey1;licenseKey2",pLicenseContent)
else:
   initLicenseFromServer("","licenseKey1;licenseKey2")
   #If you wish to use SDK offline, store the license information as .txt or other format
   content=outputLicenseToString()
   with open("license.txt","w") as f:
      f.write(content)
```

Note:

- The license verification process on the development machine can be a one-time process. Once it is registered, the registration file for this specific device can be returned and stored to the machine.
- If you need to increase the quota of your existing license key, please contact us.

Option2: Always connect to Dynamsoft server for license verification

If your development machine can access Internet all the time, you can use the `initLicenseFromServer()` method to register the development license. It will connect to Dynamsoft server for license verification each time you use the SDK.

Code snippet in c:

```
void* _br = NULL;
int iLicMsg = -1;
_br = DBR_CreateInstance();
// Connect to the Dynamsoft server to verify the license
iLicMsg = DBR_InitLicenseFromServer(_br, "", "licenseKey1;licenseKey2");
// If error occurs to the license
if (iLicMsg != DBR_OK) {
   printf("Failed to initialize the license: %d\r\n%s\r\n", iLicMsg, DBR_GetErrorString(iLicMsg));
   return iLicMsg;
}
// Barcode decoding happens here
//....
DBR_DestroyInstance(_br);
```

Code snippet in cpp:

```
CBarcodeReader* reader = new CBarcodeReader();
int iLicMsg = -1;
// Connect to Dynamsoft server to verify the license.
iLicMsg = reader->InitLicenseFromServer("", "licenseKey1;licenseKey2");
// If error occurs to the license
if (iLicMsg != DBR_OK)
{
   printf("Failed to initialize the license: %d\r\n%s\r\n", iLicMsg, CBarcodeReader::GetErrorString(iLicMsg));
   return iLicMsg;
}
// Decode barcodes happens here
//....
delete reader;
```

Code snippet in csharp:

```
int iLicMsg = -1;
```

```
BarcodeReader _br = new BarcodeReader();
// Connect to Dynamsoft server to verify the license.
iLicMsg = _br.InitLicenseFromServer("", "licenseKey1;licenseKey2");
if(iLicMsg != 0)
{
   Console.WriteLine("License error Code:",iLicMsg);
   return;
}
// Decode barcodes happens here
//....
_br.Dispose();
```

Code snippet in java:

```
try {
   BarcodeReader reader = new BarcodeReader();
   // Connect to Dynamsoft server to verify the license.
   reader.initLicenseFromServer("", "licenseKey1;licenseKey2");
   //now you are free to start barcode decoding.
}
catch(Exception ex) {
   //if your license is invalid, a BarcodeReaderException will be throw out
   //com.dynamsoft.barcode.BarcodeReaderException: The license key is invalid.
   System.out.println(ex);
}
```

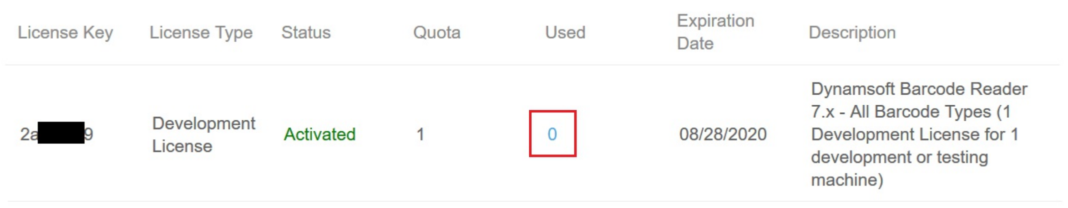
Code snippet in py:

```
def initLicenseFromServer(pLicenseServer,pLicenseKey):
    dbr.initLicenseFromServer(pLicenseServer,pLicenseKey)
#Check if there is a license file on the local machine. If not, connect to Dynamsoft Hosted server to verify the license. Otherwise, use the local license file.
if os.path.exists(license.txt):
   #Use the local license file to activate the SDK
   with open("license.txt","r") as f:
      pLicenseContent = f.read()
   initLicenseFromLicenseContent("licenseKey1;licenseKey2",pLicenseContent)
else:
   initLicenseFromServer("","licenseKey1;licenseKey2")
   #If you wish to use SDK offline, store the license information as .txt or other format
   content = outputLicenseToString()
   with open("license.txt","w") as f:
      f.write(content)
```

Option3: No Internet connection

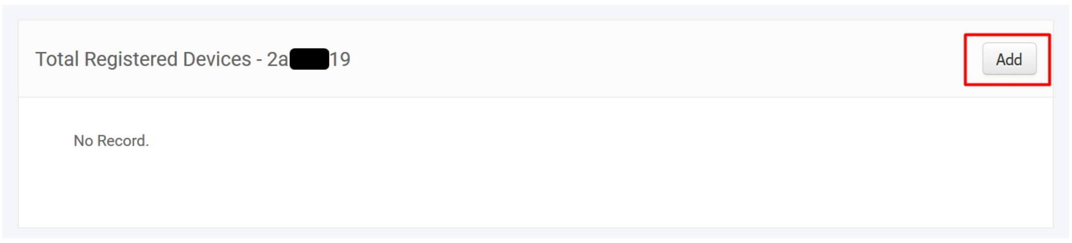
If your machine is not allowed to access Internet, you can follow the steps below to manually register the device and get the license content.

1. Log in **Customer Portal** and go to **License Center** > **Barcode Reader SDK**. Click the number under **Used**.



License Key	License Type	Status	Quota	Used	Expiration Date	Description
2a[redacted]9	Development License	Activated	1	0	08/28/2020	Dynamsoft Barcode Reader 7.x - All Barcode Types (1 Development License for 1 development or testing machine)

2. Click the **Add** button to add a device.

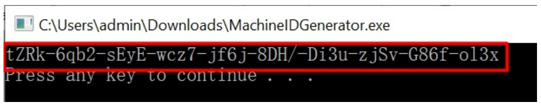

Total Registered Devices - 2a[redacted]19

No Record.

3. Get amd run the Dynamsoft tool on the device to be registered and get the machine ID.

   For Windows:

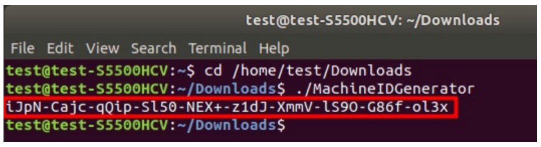
   Download MachineIDGenerator.exe and run it. The returned string, e.g. `tZRk-6qb2-sEyE-wcz7-jf6j-8DH/-Di3u-zjSv-G86f-ol3x`, is the machine ID.



```
C:\Users\admin\Downloads\MachineIDGenerator.exe
tZRk-6qb2-sEyE-wcz7-jf6j-8DH/-Di3u-zjSv-G86f-ol3x
Press any key to continue . . .
```

   For Linux:

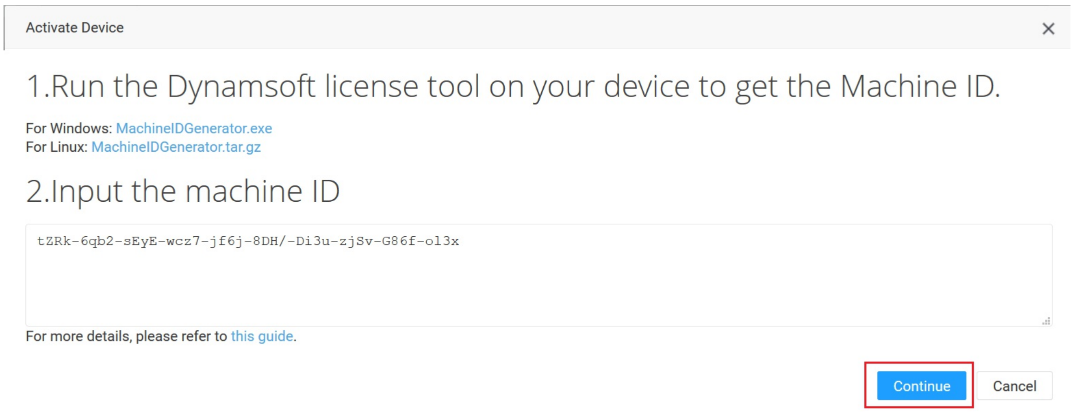
   Download MachineIDGenerator.tar.gz and unzip it. Open **Terminal** and type `./MachineIDGenerator`. The returned string, e.g. `iJpN-Cajc-qQip-Sl50-NEX+-z1dJ-XmmV-lS90-G86f-ol3x`, is the machine ID.



test@test-S5500HCV: ~/Downloads
File Edit View Search Terminal Help

```
test@test-S5500HCV:~$ cd /home/test/Downloads
test@test-S5500HCV:~/Downloads$ ./MachineIDGenerator
iJpN-Cajc-qQip-Sl50-NEX+-z1dJ-XmmV-lS90-G86f-ol3x
test@test-S5500HCV:~/Downloads$
```

4. Input the machine ID in the text box and click **Continue**.



Activate Device

1.Run the Dynamsoft license tool on your device to get the Machine ID.

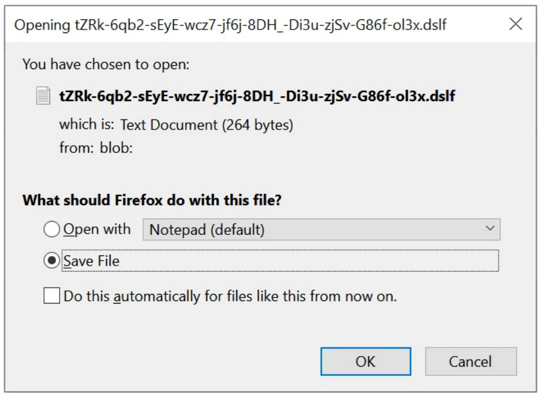
For Windows: MachineIDGenerator.exe
For Linux: MachineIDGenerator.tar.gz

2.Input the machine ID

tZRk-6qb2-sEyE-wcz7-jf6j-8DH/-Di3u-zjSv-G86f-ol3x

For more details, please refer to this guide.

   Then the license file (`.dslf`), which contains the license content, will be downloaded automatically.



Opening tZRk-6qb2-sEyE-wcz7-jf6j-8DH_-Di3u-zjSv-G86f-ol3x.dslf

You have chosen to open:

tZRk-6qb2-sEyE-wcz7-jf6j-8DH_-Di3u-zjSv-G86f-ol3x.dslf

which is: Text Document (264 bytes)
from: blob:

What should Firefox do with this file?

Open with Notepad (default) [✓ Save File]

☐ Do this automatically for files like this from now on.

5. Use the `initLicenseFromLicenseContent(licenseKey, licenseContent)` API to activate the SDK offline.

   `licenseKey` : 8-digit key in the customer portal `licenseContent` : the string in the `.dslf` file

Code snippet in c:

```
void* _br = NULL;
int iLicMsg = -1;
_br = DBR_CreateInstance();
// Use the SDK offline
iLicMsg = DBR_InitLicenseFromLicenseContent(_br, "licenseKey1;licenseKey2", "LicenseContent"));
// If error occurs to the license
if (iLicMsg != DBR_OK) {
   printf("Failed to initialize the license: %d\r\n%s\r\n", iLicMsg, DBR_GetErrorString(iLicMsg));
   return iLicMsg;
}
// Barcode decoding happens here
//....
DBR_DestroyInstance(_br);
```

Code snippet in cpp:

```
CBarcodeReader* reader = new CBarcodeReader();
int iLicMsg = -1;
// Use the SDK offline
iLicMsg = reader->InitLicenseFromLicenseContent("licenseKey1;licenseKey2", "LicenseContent");
// If error occurs to the license
if (iLicMsg != DBR_OK)
{
   printf("Failed to initialize the license: %d\r\n%s\r\n", iLicMsg, CBarcodeReader::GetErrorString(iLicMsg));
   return iLicMsg;
}
// Decode barcodes happens here
//....
delete reader;
```

Code snippet in csharp:

```
int iLicMsg = -1;
BarcodeReader _br = new BarcodeReader();
// Use the SDK offline
iLicMsg = _br.InitLicenseFromLicenseContent("licenseKey1;licenseKey2", "LicenseContent");
if(iLicMsg != 0)
{
   Console.WriteLine("License error Code:",iLicMsg);
   return;
}
// Decode barcodes happens here
//....
_br.Dispose();
```

Code snippet in java:

```
try{
   BarcodeReader reader = new BarcodeReader();
   // Use the SDK offline
   reader.initLicenseFromLicenseContent("", "licenseKey1;licenseKey2", "LicenseContent");
}catch(Exception e) { //if your license is invalid, a BarcodeReaderException will be throw out
   System.out.println(e);
}
```

Code snippet in py:

```
def initLicenseFromLicenseContent(plicenseKey, pLicenseContent):
   dbr.initLicenseFromLicenseContent(plicenseKey, pLicenseContent)
```

If you run into any issues, please contact Dynamsoft Support.

Use a Runtime License

There are two types of license keys, the 8-digit short key and the long product key (e.g., `f0068MgAAAKUqfMMSFXqnKZG6nkjs4x1*******************************zxsbWrrPLtAFXQ0jSutuZ7M=`).

For short 8-digit license key

The procedure for setting a short runtime license is the same as using a development license. You can use `initLicenseFromServer()` or `initLicenseFromLicenseContent()` to complete the license registration. Please refer to Use a Development License section for more information.

For long license key

If you are using a long key, please refer to Use a Trial License section for more information.

Distribution

Licensing

Once you finish the development and testing of your application with a Development License, you'll need a Runtime License to distribute your application. For more information on how to set a runtime license, please refer to Use a Runtime License.

Distributing

Distribute the appropriate assembly files with the application using Dynamsoft Barcode Reader SDK.

Programming language	Files for distribution or deployment
C/C++	all files under `Dynamsoft\BarcodeReader\lib\`
Java	`dynamsoft-barcodereader-{version number}.jar`

Note: If you want to read barcodes from PDF files through C/C++ API, please put `libDynamicPdf.so` in the same folder as the core assembly file - `libDynamsoftBarcodeReader.so` . Otherwise, this is not required.

License Errors

Symptoms

When you scan barcodes, you may get barcode results marked with asterisks (*), like this:

> 123456789* {barcode type} barcode license invalid, please contact support@dynamsoft.com to get a valid trial license.

Solutions

If you are using trial version, it indicates that your license has expired. You can log in the customer portal and request for a trial extension online.

If you are using full version, please check the error code in order to troubleshoot the issue. Below is the code snippet shows how to get the error code:

Code snippet in c:

```
void* _br = NULL;
int iLicMsg = -1;
_br = DBR_CreateInstance();
// Connect to the Dynamsoft server to verify the license
iLicMsg = DBR_InitLicenseFromServer(_br, "", "licenseKey1;licenseKey2");
// If error occurs to the license
if (iLicMsg != DBR_OK) {
   printf("Failed to initialize the license: %d\r\n%s\r\n", iLicMsg, DBR_GetErrorString(iLicMsg));
   return iLicMsg;
}
```

Code snippet in cpp:

```
CBarcodeReader* reader = new CBarcodeReader();
int iLicMsg = -1;
// Connect to Dynamsoft server to verify the license.
iLicMsg = reader->InitLicenseFromServer("", "licenseKey1;licenseKey2");
// If error occurs to the license
if (iLicMsg != DBR_OK)
{
   printf("Failed to initialize the license: %d\r\n%s\r\n", iLicMsg, CBarcodeReader::GetErrorString(iLicMsg));
   return iLicMsg;
}
```

Code snippet in csharp:

```
int iLicMsg = -1;
BarcodeReader _br = new BarcodeReader();
// Connect to Dynamsoft server to verify the license.
iLicMsg = _br.InitLicenseFromServer("", "licenseKey1;licenseKey2");
if(iLicMsg != 0)
{
   Console.WriteLine("License error Code:", iLicMsg);
   return;
}
```

Code snippet in java:

```
 int iLicMsg = -1
```

```
try{
    BarcodeReader reader = new BarcodeReader();
    BarcodeReader.initLicenseFromServer("", "licenseKey1;licenseKey2");
}
catch (BarcodeReaderException e) {
    iLicMsg = e.getErrorCode();
    String pszTemp = String.format(" Error Code %d:", iLicMsg);
    System.out.println(pszTemp);
    pszTemp = " Error Message: " + e.getMessage();
    System.out.println(pszTemp);
}
```

Code snippet in python:

```
    def InitLicense(license):
    errorCode = dbr.InitLicense(license)
    print("Error Code: ")
    print(errorCode)
```

Error Code

- **Error Code: -10044**

  Error Message: Failed to request the license file

  Solution:

  Check your network connection and make sure you have Internet access. If you have a firewall configured on the device, it is very likely that our license server is blocked by your firewall. Please contact Dynamsoft to resolve the issue.

- **Error Code: -10054**

  Error Message: The device number in the license key runs out

  Solution:

  You can contact Dynamsoft to expand the volume of your current runtime license key. Rest assured that your license key remains unchanged during the upgrade process, so no code change is required to your application.

- **Error Code: -10004**

  Error Message: The license has expired

  Solution:

  Your annual runtime license has expired. You can log into the customer portal to renew your runtime license by credit card or PayPal. Alternatively, you can contact Dynamsoft if you prefer other payment methods (wire transfer or check). Rest assured that your license key remains unchanged during the upgrade process, so no code change is required to your application.

- **Error Code: -10042**

  Error Message: The license DLL is missing (for C/C++)

  Solution:

  For 8-digit license keys, we use a separate license DLL to verify the License. Please copy `DynamsoftLicClientx64.dll` (or `DynamsoftLicClientx86.dll`) from `[INSTALLATION FOLDER]\Components\C_C++\Redist\x64\` (or `[INSTALLATION FOLDER]\Components\C_C++\Redist\x86`) to the same folder as the barcode reader dll `DynamsoftBarcodeReaderx64.dll` (or `DynamsoftBarcodeReaderx86.dll`).

- **Error Code: -10052**

  Error Message: The license content is invalid

Solution:

This error happens when you are trying to use InitLicenseFromLicenseContent() API to activate the license. Please refer to Use a Development License section to double check if the license content is correct.

If this article doesn't help resolve the license issue, please contact Dynamsoft for further assistance.

FAQ

Licensing

How to find my license key for the full version?

You can find the license key from the license email. It is also available under the “License Center” -> “Barcode Reader SDK” section inside your Dynamsoft account.

Does Dynamsoft Barcode Scanner SDK require an Internet connection?

For web applications, it doesn’t require any Internet connection.

For mobile apps, the device must go online to complete the device registration for the first time of using the barcode scanning feature. Afterwards, the mobile device can work offline until the current runtime license key expires.

For desktop applications, an Internet connection is required the first time the device opens the barcode scanner (i.e. the InitLicenseFromServer() method is executed). After the device connects to our license server successfully, you can use the OutputLicenseToString API to get the information of the license and store the information to the device. Afterwards, you can use initLicenseFromServerContent API to use the SDK in offline mode. For more information, please refer to this article.

When does a device counts as an activated device?

Invoking InitLicenseFromServer() method automatically activated the device.

For an environment with no internet connection allowed, can I use your barcode reader SDK?

Yes. You can follow the instruction here to manually register the device. For enterprise customers who can't manually register devices, we offer a few flexible options, please contact Dynamsoft for details.

Using Barcode Reader

When I scan barcodes, why are the barcode results marked with asterisks (*)?

Your trial license has expired or you don’t have a valid license key included in the code. You can log into the customer portal to generate a free 30-day trial license or purchase a full license on the Online Store.

I saw you have Windows Edition, Linux Edition, JavaScript Edition, etc. Which is the right one for my application?

Check out Dynamsoft Barcode Reader SDK Edition Comparison to identify the right product/edition.

Why does it return strange characters (messy code, gibberish, or non-printable) as a result?

Some barcodes are encoded with non-printable characters (such as \0) or other different types (UTF-16, GB2312, etc.). We are using UTF-8 encoding type in our sample applications for demonstration purposes. In your application, you need to use BarcodeBytes to get the raw data and then convert it to the desired encoding instead of using BarcodeText directly.

How to get the “result image” with overlays once barcodes are found in the image?

It is possible to get the resulting image with overlays. When a barcode is found, our library will not only return the barcode text, but also the coordinates of the barcode. You can add a rectangle on the barcode to highlight it so that users know which barcode is being scanned.

The barcode reader SDK sometimes return false results with four or fewer characters. How to avoid it?

You may get results with four or fewer characters for Industrial_25 and ITF barcode symbologies. This is because these two symbologies have weak error protection and checksum. To avoid such cases, you can try:

- Use MinResultConfidence to filter out barcode results having Confidence of less than 35. The Confidence tells us how confident it is about the decoding result. The value ranges from 0 (uncertain) to 100 (100% correct).
- Use MinBarcodeTextLength to set a minimum character length for the barcode results. For more information to filter out unwanted barcode results, please refer to this article.

Can I scan barcodes on US Driver’s Licenses?

Yes, the barcode on US Driver’s Licenses is a PDF417 code and can be scanned with our library. The result is returned in raw data and you’ll need to parse it into human-readable formats.

Additional resources

Demos and samples

- Read barcodes from images online demo
- Code gallery
- Github repositories

API reference

- Windows Edition
- Linux Edition
- JavaScript Edition
- iOS Edition
- Android Edition

Release notes

Release Notes - Dynamsoft Barcode Reader SDK

Get support

If you need any help, please feel free to contact us via email, telephone, live chat etc.

Contact Dynamsoft >

How to upgrade from a trial to full version

- For Version 6.5+ and 7.x
- For Version 6.4.1 and earlier

For Version 6.5+ and 7.x

For development and testing, please refer to the Set Development License section.

For deployment and distribution, please refer to the Set Runtime License and Distribution sections.

For Version 6.4.1 and earlier

To upgrade Dynamsoft Barcode Reader from trial version to licensed version, you only need to replace the license key. No other value needs to be changed.

Please search for "Licensekey" or "InitLicense" in the code and replace the trial license with the full one.

How to upgrade to a newer version

- Upgrade from version 7.x to 7.x
- Upgrade from version 6.x to 7.x

Upgrade from version 7.x to 7.x

For minor upgrade between 7.x, you only need to replace the old assembly files with the new ones. Your previous SDK license is still compatible with the new version. See Distribution section for more info.

Upgrade from version 6.x to 7.x

We made some structural updates in v7.0. To upgrade from 6.x to 7.x, we recommend you to review our sample code below to upgrade your barcode reading module using the latest version of our SDK.

API changes

- V7.x C/C++ API changes
- V7.x .NET API changes
- V7.x Java API changes
- V7.x API changes in iOS Edition

Runtime settings changes

Comparing to version 6.x, some runtime settings have been removed while many new runtime settings have been added to version 7.x. And in v7.x, to use the runtime settings with arguments, you can use the method `SetModeArgument()` .

- Localization algorithm
- Image binarization control
- The texture filter mode
- Region predetection mode
- Gray equalization sensitivity
- Texture detection sensitivity
- Localization result

Localization algorithm

In v6.x, `mLocalizationAlgorithmPriority` and `mAntiDamageLevel` are used together to control the priority and the number of localization algorithms to be used. In v7.x, such control is conducted by `localizationModes` , using the index of the array to determine the priority, and the element of the array determines the localization algorithms to be used for barcode detection. In addition, v7.x added a new algorithm,'Scan Directly'(variable name: `LM_SCAN_DIRECTLY`), which is optimized for interactive scenarios such as scanning barcode via mobile camera.

Sample code using 6.x version with C++:

```
reader.GetRuntimeSettings(&runtimeSettings);
runtimeSettings.mLocalizationAlgorithmPriority="Statistics, Lines, ConnectedBlocks, FullImageAsBarcodeZone";
runtimeSettings.mAntiDamageLevel=7;
reader.UpdateRuntimeSettings(&runtimeSettings,szErrorMsg,256);
```

Sample code using 7.x version with C++:

```
reader.GetRuntimeSettings(&runtimeSettings);
runtimeSettings.localizationModes[0] = LM_STATISTICS;
runtimeSettings.localizationModes[1] = LM_LINES;
runtimeSettings.localizationModes[2] = LM_CONNECTED_BLOCKS ;
runtimeSettings.localizationModes[3] = LM_SKIP;
reader.UpdateRuntimeSettings(&runtimeSettings,szErrorMsg,256);
```

Image binarization control

In v6.x, binarization is controlled by the runtime settings, `mEnableFillBinaryVacancy` and `mBinarizationBlockSize`. In v7.x, such process is set and controlled by `binarizationModes`. Such mode can be set to one or more values. In addition, the `binarizationModes` contains many arguments, including `EnableFillBinaryVacancy`, `BlockSizeX`, `BlockSizeY`.

Sample code using 6.x version with C++:

```
reader.GetRuntimeSettings(&runtimeSettings);
runtimeSettings.mEnableFillBinaryVacancy= 1;
runtimeSettings.mBinarizationBlockSize= 9;
reader.UpdateRuntimeSettings(&runtimeSettings,szErrorMsg,256);
```

Sample code using 7.x version with C++:

```
reader.GetRuntimeSettings(&runtimeSettings);
runtimeSettings.binarizationModes.[0] = BM_LOCAL_BLOCK;
reader.UpdateRuntimeSettings(&runtimeSettings,szErrorMsg,256);
reader.SetModeArgument("BinarizationModes",0, "EnableFillBinaryVacancy","1");
reader.SetModeArgument("BinarizationModes",0, "BlockSizeX","9");
```

The texture filter mode

In v6.x, the texture filter mode (`mTextFilterMode`) can only be set to `Enable` or `Disable`. In v7.x, the text filter(`textFilterModes`) can be set to one or more modes using `PublicRuntimeSettings.furtherModes.textFilterModes`. It contains two arguments including `MinImageDimension` and `Sensitivity`.

Sample code using 6.x version with C++:

```
reader.GetRuntimeSettings(&runtimeSettings);
runtimeSettings.mTextFilterMode= TFM_Enable;
reader.UpdateRuntimeSettings(&runtimeSettings,szErrorMsg,256);
```

Sample code using 7.x version with C++:

```
reader.GetRuntimeSettings(&runtimeSettings);
runtimeSettings.furtherModes.textFilterModes[0] = TFM_GENERAL_CONTOUR;
reader.UpdateRuntimeSettings(&runtimeSettings,szErrorMsg,256);
reader.SetModeArgument("TextFilterModes",0, "Sensitivity","5");
```

Region predetection mode

In v6.x, the region predetection (`mRegionPredetectionMode`) can only be set to `Enable` or `Disable`. In v7.x, the region predetection (`regionPredetectionModes`) can be set to one or more modes using `PublicRuntimeSettings.furtherModes.regionPredetectionModes`. It contains two arguments including `MinImageDimension` and `Sensitivity`.

Sample code using 6.x version with C++:

```
reader.GetRuntimeSettings(&runtimeSettings);
runtimeSettings.mRegionPredetectionMode= RPM_Enable ;
reader.UpdateRuntimeSettings(&runtimeSettings,szErrorMsg,256);
```

Sample code using 7.x version with C++:

```
reader.GetRuntimeSettings(&runtimeSettings);
runtimeSettings.furtherModes.regionPredetectionModes[0] = RPM_GENERAL;
reader.UpdateRuntimeSettings(&runtimeSettings,szErrorMsg,256);
reader.SetModeArgument("regionPredetectionModes",0, "Sensitivity","5");
```

Gray equalization sensitivity

In v6.x, the gray equalization sensitivity (`mGrayEqualizationSensitivity`) is used to set the sensitivity for gray equalization. In v7.x, sensitivity is used as an argument of `IPM_GRAY_EQUALIZE` , that's to say, the `IPM_GRAY_EQUALIZE` needs to be set in `imagePreprocessingModes` , and then use `SetModeArgument` set the sensitivity.

Sample code using 6.x version with C++:

```
reader.GetRuntimeSettings(&runtimeSettings);
runtimeSettings.mGrayEqualizationSensitivity = 9;
reader.UpdateRuntimeSettings(&runtimeSettings,szErrorMsg,256);
```

Sample code using 7.x version with C++:

```
reader.GetRuntimeSettings(&runtimeSettings);
runtimeSettings.furtherModes.imagePreprocessingModes[0] = IPM_GENERAL;
runtimeSettings.furtherModes.imagePreprocessingModes[1] = IPM_GRAY_EQUALIZE;
reader.UpdateRuntimeSettings(&runtimeSettings,szErrorMsg,256);
reader.SetModeArgument("ImagePreprocessingModes ",1, "Sensitivity","9");
```

Texture detection sensitivity

In v6.x, the texture detection sensitivity (`mTextureDetectionSensitivity`) is used to set the sensitivity for texture detection, to smooth and filter out the texture of the background such as the texture of the screen when one tries to scan barcodes on screen. In v7.x, sensitivity is used as an argument of texture detection.

Sample code using 6.x version with C++:

```
reader.GetRuntimeSettings(&runtimeSettings);
runtimeSettings.mTextureDetectionSensitivity = 9;
reader.UpdateRuntimeSettings(&runtimeSettings,szErrorMsg,256);
```

Sample code using 7.x version with C++:

```
reader.GetRuntimeSettings(&runtimeSettings);
runtimeSettings.furtherModes.textureDetectionModes[0] = TDM_GENERAL_WIDTH_CONCENTRATION;
reader.UpdateRuntimeSettings(&runtimeSettings,szErrorMsg,256);
reader.SetModeArgument("textureDetectionModes",0, "Sensitivity","9");
```

Localization result

In v6.x, all the localization results can be obtained by the method `GetAllLocalizationResults()` , including the localized but not decoded barcodes. In v7.x, only the results of the decoded barcodes can be stored in `LocalizationResult` . To obtain all the localized barcodes including not decoded ones, you need to set `IRT_TYPED_BARCODE_ZONE` .

Sample code using 7.x version with C++:

```
reader.GetRuntimeSettings(&runtimeSettings);
runtimeSettings.intermediateResultTypes = IRT_TYPED_BARCODE_ZONE;
reader.UpdateRuntimeSettings(&runtimeSettings,szErrorMsg,256);
reader.DecodeFile("Your image file path", "");
IntermediateResultArray* pResults;
reader.GetIntermediateResults(&pResults);
for(int irIndex=0; irIndex<pIntermediateResults->resultsCount; irIndex++)
```

```
{
    IntermediateResult* ir = pIntermediateResults->results[irIndex];
    for(int lrIndex=0; lrIndex<ir->resultsCount; lrIndex++)
    {
        LocalizationResult* lr = (LocalizationResult*) ir->results[lrIndex];
        //your codes to using the localization result
    }
}
```

V7.x C/C++ API changes

Comparing to the version 6.x, version 7.x has removed some APIs and functions and renamed many APIs, variables, and struct. The C/C++ API changes are listed below:

APIs removed

C	C++
DBR_GetAllLocalizationResults	CBarcodeReader::GetAllLocalizationResults
DBR_FreeLocalizationResults	CBarcodeReader::FreeLocalizationResults
DBR_LoadSettingsFromFile	CBarcodeReader::LoadSettingsFromFile
DBR_LoadSettings	CBarcodeReader::LoadSettings
DBR_AppendParameterTemplateFromFile	CBarcodeReader::AppendParameterTemplateFromFile
DBR_AppendParameterTemplate	CBarcodeReader::AppendParameterTemplate
DBR_GetTemplateSettings	CBarcodeReader::GetTemplateSettings
DBR_SetTemplateSettings	CBarcodeReader::SetTemplateSettings

SExtendedResult renamed to ExtendedResult

Version 6.x	Version 7.x
emResultType	resultType
emBarcodeFormat	barcodeFormat
pszBarcodeFormatString	barcodeFormatString
iConfidence	confidence
pBytes	bytes
nBytesLength	bytesLength

SLocalizationResult renamed to LocalizationResult

Extended results info is moved to struct TextResult.

Version 6.x	Version 7.x
emTerminateStage	terminatePhase
emBarcodeFormat	barcodeFormat
pszBarcodeFormatString	barcodeFormatString
iX1	x1
iY1	y1
iX2	x2
iY2	y2
iX3	x3

iY3	y3
iX4	x4
iY4	y4
iAngle	angle
iModuleSize	moduleSize
iPageNumber	pageNumber
pszRegionName	regionName
pszDocumentName	documentName
nResultsCount	Removed
ppResults	Removed

STextResult renamed to TextResult

Version 6.x	Version 7.x
emBarcodeFormat	barcodeFormat
pszBarcodeFormatString	barcodeFormatString
pszBarcodeText	barcodeText
pBarcodeBytes	barcodeBytes
nBarcodeBytesLength	barcodeBytesLength
pLocalizationResult	localizationResult

STextResultArray renamed to TextResultArray

Version 6.x	Version 7.x
nResultsCount	resultsCount
ppResults	results

Renamed or removed members for PublicRuntimeSettings

Version 6.x	Version 7.x
mTimeout	timeout
mPDFRasterDPI	pdfRasterDPI
mBarcodeFormatIds	barcodeFormatIds
mMaxAlgorithmThreadCount	maxAlgorithmThreadCount
mDeblurLevel	deblurLevel
mScaleDownThreshold	scaleDownThreshold
mExpectedBarcodesCount	expectedBarcodesCount
mTextFilterMode	Removed
mRegionPredetectionMode	Removed
mLocalizationAlgorithmPriority	Removed
mTextureDetectionSensitivity	Removed
mAntiDamageLevel	Removed
mMaxDimOfFullImageAsBarcodeZone	Removed
mMaxBarcodesCount	Removed
mBarcodeInvertMode	Removed
mGrayEqualizationSensitivity	Removed

mEnableFillBinaryVacancy	Removed
mColourImageConvertMode	Removed
mBinarizationBlockSize	Removed

Other renamed or removed enumerations or structs

Version 6.x	Version 7.x
SLocalizationResultArray	Removed
PublicParameterSettings	Removed
enum BarcodeInvertMode	Removed
enum ColourImageConvertMode	Removed
enum TerminateStage	Removed
enum RegionPredetectionMode	Removed
enum TextFilterMode	Removed
BarcodeFormat	
BF_All	BF_ALL
BF_OneD	BF_ONED
ResultType	
EDT_StandardText	RT_STANDARD_TEXT
EDT_RawText	RT_RAW_TEXT
EDT_CandidateText	RT_CANDIDATE_TEXT
EDT_PartialText	RT_PARTIAL_TEXT
enum ConflictMode	
ECM_Ignore	CM_IGNORE
ECM_Overwrite	CM_OVERWRITE
ImagePixelFormat	
IPF_Binary	IPF_BINARY
IPF_BinaryInverted	IPF_BINARY_INVERTED
IPF_GrayScaled	IPF_GRAYSCALED

V7.x .NET API changes

Comparing to the version 6.x, version 7.x has removed some APIs and functions and renamed many APIs, variables, and struct. The changes for the .NET APIs are listed below:

APIs removed

.NET
GetAllLocalizationResults
SetTemplateSettings
GetTemplateSettings
LoadSettings
LoadSettingsFromFile
AppendParameterTemplate
AppendParameterTemplateFromFile

Renamed or removed members for PublicRuntimeSettings

Version 6.x	Version 7.x
mTimeout	Timeout
mPDFRasterDPI	PDFRasterDPI
mBarcodeFormatIds	BarcodeFormatIds
mMaxAlgorithmThreadCount	MaxAlgorithmThreadCount
mDeblurLevel	DeblurLevel
mScaleDownThreshold	ScaleDownThreshold
mExpectedBarcodesCount	ExpectedBarcodesCount
mTextFilterMode	Removed
mRegionPredetectionMode	Removed
mLocalizationAlgorithmPriority	Removed
mTextureDetectionSensitivity	Removed
mAntiDamageLevel	Removed
mMaxDimOfFullImageAsBarcodeZone	Removed
mMaxBarcodesCount	Removed
mBarcodeInvertMode	Removed
mGrayEqualizationSensitivity	Removed
mEnableFillBinaryVacancy	Removed
mColourImageConvertMode	Removed
mBinarizationBlockSize	Removed

Renamed or removed class members

Extended results info is now moved to Class `TextResult` .

Version 6.x	Version 7.x
Class Localization Result	
TerminateStage	TerminatePhase
ExtendedResultArray	Removed
Class BarcodeReader	
LicenseKeys	ProductKeys

Renamed barcode format

Version 6.x	Version 7.x
BF_All	BF_ALL
BF_OneD	BF_ONED
CODE_39	BF_CODE_39
CODE_128	BF_CODE_128
CODE_93	BF_CODE_93
CODABAR	BF_CODABAR
ITF	BF_ITF
EAN_13	BF_EAN_13
EAN_8	BF_EAN_8
UPC_A	BF_UPC_A
UPC_E	BF_UPC_E
INDUSTRIAL_25	BF_INDUSTRIAL_25
PDF417	BF_PDF417
QR_CODE	BF_QR_CODE
DATAMATRIX	BF_DATAMATRIX
AZTEC	BF_AZTEC

Renamed EnumResultType

Version 6.x	Version 7.x
EDT_StandardText	RT_STANDARD_TEXT
EDT_RawText	RT_RAW_TEXT
EDT_CandidateText	RT_CANDIDATE_TEXT
EDT_PartialText	RT_PARTIAL_TEXT

Renamed EnumConflict Mode

Version 6.x	Version 7.x
ECM_Ignore	CM_IGNORE
ECM_Overwrite	CM_OVERWRITE

Renamed EnumErrorCode

Version 6.x	Version 7.x
DBR_AZTEC_LICENSE_INVALID	DBRERR_AZTEC_LICENSE_INVALID
DBR_PARAMETER_VALUE_INVALID	DBRERR_PARAMETER_VALUE_INVALID
DBR_JSON_NAME_REFERENCE_INVALID	DBRERR_JSON_NAME_REFERENCE_INVALID
DBR_TEMPLATE_NAME_INVALID	DBRERR_TEMPLATE_NAME_INVALID
DBR_JSON_NAME_VALUE_DUPLICATED	DBRERR_JSON_NAME_VALUE_DUPLICATED
DBR_JSON_NAME_KEY_MISSING	DBRERR_JSON_NAME_KEY_MISSING
DBR_JSON_VALUE_INVALID	DBRERR_JSON_VALUE_INVALID
DBR_JSON_KEY_INVALID	DBRERR_JSON_KEY_INVALID
DBR_JSON_TYPE_INVALID	DBRERR_JSON_TYPE_INVALID
DBR_JSON_PARSE_FAILED	DBRERR_JSON_PARSE_FAILED
DBR_RECOGNITION_TIMEOUT	DBRERR_RECOGNITION_TIMEOUT
DBR_CUSTOM_MODULESIZE_INVALID	DBRERR_CUSTOM_MODULESIZE_INVALID
DBR_CUSTOM_SIZE_INVALID	DBRERR_CUSTOM_SIZE_INVALID
DBR_PAGE_NUMBER_INVALID	DBRERR_PAGE_NUMBER_INVALID
DBR_PDF_DLL_MISSING	DBRERR_PDF_DLL_MISSING
DBR_PDF_READ_FAILED	DBRERR_PDF_READ_FAILED
DBR_DATAMATRIX_LICENSE_INVALID	DBRERR_DATAMATRIX_LICENSE_INVALID
DBR_PDF417_LICENSE_INVALID	DBRERR_PDF417_LICENSE_INVALID
DBR_DIB_BUFFER_INVALID	DBRERR_DIB_BUFFER_INVALID
DBR_1D_LICENSE_INVALID	DBRERR_1D_LICENSE_INVALID
DBR_QR_LICENSE_INVALID	DBRERR_QR_LICENSE_INVALID
DBR_TIFF_READ_FAILED	DBRERR_TIFF_READ_FAILED
DBR_IMAGE_READ_FAILED	DBRERR_IMAGE_READ_FAILED
DBR_MAX_BARCODE_NUMBER_INVALID	DBRERR_MAX_BARCODE_NUMBER_INVALID
DBR_CUSTOM_REGION_INVALID	DBRERR_CUSTOM_REGION_INVALID
DBR_BARCODE_FORMAT_INVALID	DBRERR_BARCODE_FORMAT_INVALID
DBR_INDEX_INVALID	DBRERR_INDEX_INVALID
DBR_BPP_NOT_SUPPORTED	DBRERR_BPP_NOT_SUPPORTED
DBR_FILETYPE_NOT_SUPPORTED	DBRERR_FILETYPE_NOT_SUPPORTED
DBR_FILE_NOT_FOUND	DBRERR_FILE_NOT_FOUND
DBR_LICENSE_EXPIRED	DBRERR_LICENSE_EXPIRED
DBR_LICENSE_INVALID	DBRERR_LICENSE_INVALID
DBR_NULL_REFERENCE	DBRERR_NULL_POINTER
DBR_NO_MEMORY	DBRERR_NO_MEMORY
DBR_UNKNOWN	DBRERR_UNKNOWN

Other renamed or removed enumerations

Version 6.x	Version 7.x
BarcodeInvertMode	Removed
ColourImageConvertMode	Removed
EnumTerminateStage	Removed
EnumImagePixelFormat	
IPF_Binary	IPF_BINARY

IPF_BinaryInverted	IPF_BINARY_INVERTED
IPF_GrayScaled	IPF_GRAYSCALED
RegionPredetectionMode	
RPM_Disable	RPM_Enable
TextFilterMode	
TFM_Disable	TFM_Enable

V7.x Java API changes

Comparing to the version 6.x, version 7.x has removed some APIs and functions and renamed many API, variables and struct. The changes for the Java APIs are listed below:

APIs removed

Java
getAllTextResults
getAllLocalizationResults
setTemplateSettings
getTemplateSettings
loadSettings
loadSettingsFromFile
appendParameterTemplate
appendParameterTemplateFromFile

Renamed or removed members for PublicRuntimeSettings

Version 6.x	Version 7.x
mTimeout	timeout
mPDFRasterDPI	pdfRasterDPI
mBarcodeFormatIds	barcodeFormatIds
mMaxAlgorithmThreadCount	maxAlgorithmThreadCount
mDeblurLevel	deblurLevel
mScaleDownThreshold	scaleDownThreshold
mExpectedBarcodesCount	expectedBarcodesCount
mTextFilterMode	Removed
mRegionPredetectionMode	Removed
mLocalizationAlgorithmPriority	Removed
mTextureDetectionSensitivity	Removed
mAntiDamageLevel	Removed
mMaxDimOfFullImageAsBarcodeZone	Removed
mMaxBarcodesCount	Removed
mBarcodeInvertMode	Removed
mGrayEqualizationSensitivity	Removed
mEnableFillBinaryVacancy	Removed
mColourImageConvertMode	Removed
mBinarizationBlockSize	Removed

Renamed or removed class members

Extended results info is now moved to Class TextResult

Version 6.x	Version 7.x
Class Localization Result	
terminateStage	terminatePhase
extendedResultArray	Removed

Renamed barcode formats

Version 6.x	Version 7.x
BF_All	BF_ALL
BF_OneD	BF_ONED

Renamed EnumResultType

Version 6.x	Version 7.x
EDT_StandardText	RT_STANDARD_TEXT
EDT_RawText	RT_RAW_TEXT
EDT_CandidateText	RT_CANDIDATE_TEXT
EDT_PartialText	RT_PARTIAL_TEXT

Renamed EnumConflict Mode

Version 6.x	Version 7.x
ECM_Ignore	CM_IGNORE
ECM_Overwrite	CM_OVERWRITE

Other renamed or removed enumerations

Version 6.x	Version 7.x
EnumBarcodeInvertMode	Removed
EnumColourImageConvertMode	Removed
EnumTerminateStage	Removed
EnumRegionPredetectionMode	
RPM_Disable	Removed
RPM_Enable	Removed
EnumImagePixelFormat	
IPF_Binary	IPF_BINARY
IPF_BinaryInverted	IPF_BINARY_INVERTED
IPF_GrayScaled	IPF_GRAYSCALED
EnumTextFilterMode	
TFM_Disable	Removed
TFM_Enable	Removed

V7.x API changes in the iOS Edition

Comparing to the version 6.x, version 7.x has removed some APIs and functions and renamed many APIs, variables, and struct. The API changes for the iOS edition are listed below:

APIs removed

iOS
allLocalizationResults
appendParameterTemplate
appendParameterTemplateFromFile
templateSettingsWithName
loadSettings
loadSettingsFromFile
setTemplateSettings

Renamed classes and enumerations

Version 6.x	Version 7.x
PublicSettings	iPublicRuntimeSettings
LocalizationResult	iLocalizationResult
ExtendedResult	iExtendedResult
TextResult	iTextResult
BarcodeInvert	Removed
ColourImageConvert	Removed
TerminateStatus	Removed
BarcodeType	EnumBarcodeFormat

Renamed or removed members for PublicRuntimeSettings

Version 6.x PublicRuntimeSettings	Version 7.x iPublicRuntimeSettings
PDFRasterDPI	pdfRasterDPI
barcodeTypeID	barcodeFormatIds
scaleDownThreshold	scaleDownThreshold
expectedBarcodeCount	expectedBarcodesCount
textFilter	Removed
regionPredetection	Removed

localizationAlgorithmPriority	Removed
textureDetectionSensitivity	Removed
antiDamageLevel	Removed
maxDimOfFullImageAsBarcodeZone	Removed
maxBarcodeCount	Removed
barcodeInvert	Removed
grayEqualizationSensitivity	Removed
enableFillBinaryVacancy	Removed
colourImageConvert	Removed
binarizationBlockSize	Removed

Renamed or removed class members

Extended results info is now moved to Class `TextResult`

Version 6.x	Version 7.x
Class LocalizationResult	
terminateStage	terminatePhase
extendedResultArray	Removed

Renamed barcode types

Version 6.x BarcodeType	Version 7.x EnumBarcodeFormat
BarcodeTypeCODE39	EnumBarcodeFormatCODE39
BarcodeTypeCODE128	EnumBarcodeFormatCODE128
BarcodeTypeCODE93	EnumBarcodeFormatCODE93
BarcodeTypeCODABAR	EnumBarcodeFormatCODABAR
BarcodeTypeITF	EnumBarcodeFormatITF
BarcodeTypeEAN13	EnumBarcodeFormatEAN13
BarcodeTypeEAN8	EnumBarcodeFormatEAN8
BarcodeTypeUPCA	EnumBarcodeFormatUPCA
BarcodeTypeUPCE	EnumBarcodeFormatUPCE
BarcodeTypeINDUSTRIAL	EnumBarcodeFormatINDUSTRIAL
BarcodeTypePDF417	EnumBarcodeFormatPDF417
BarcodeTypeQRCODE	EnumBarcodeFormatQRCODE
BarcodeTypeDATAMATRIX	EnumBarcodeFormatDATAMATRIX
BarcodeTypeAZTEC	EnumBarcodeFormatAZTEC
BarcodeTypeONED	EnumBarcodeFormatONED
BarcodeTypeALL	EnumBarcodeFormatALL

Renamed enumeration for result type

Version 6.x ResultTextType	Version 7.x EnumResultType
ResultTextTypeStandardText	EnumResultTypeStandardText
ResultTextTypeRawText	EnumResultTypeRawText
ResultTextTypeCandidateText	EnumResultTypeCandidateText

ResultTextTypePartialText	EnumResultTypePartialText

Renamed enumerations for conflict mode

Version 6.x DBRConflictMode	Version 7.x EnumConflictMode
DBRECM_Ignore	EnumConflictModeIgnore
DBRECM_Overwrite	EnumConflictModeOverwrite

Renamed enumerations for image pixel format

Version 6.x ImagePixelType	Version 7.x EnumImagePixelFormat
ImagePixelTypeBinary	EnumImagePixelFormatBinary
ImagePixelTypeBinaryInverted	EnumImagePixelFormatBinaryInverted
ImagePixelTypeGrayScaled	EnumImagePixelFormatGrayScaled
ImagePixelTypeNV21	EnumImagePixelFormatNV21
ImagePixelTypeRGB_565	EnumImagePixelFormatRGB_565
ImagePixelTypeRGB_555	EnumImagePixelFormatRGB_555
ImagePixelTypeRGB_888	EnumImagePixelFormatRGB_888
ImagePixelTypeARGB_8888	EnumImagePixelFormatARGB_8888
ImagePixelTypeRGB_161616	EnumImagePixelFormatRGB_161616
ImagePixelTypeARGB_16161616	EnumImagePixelFormatARGB_16161616

Renamed enumerations for error code

Version 6.x DBRErrorCode	Version 7.x EnumErrorCode
DBRErrorCode_Unknown	EnumErrorCode_Unknown
DBRErrorCode_No_Memory	EnumErrorCode_No_Memory
DBRErrorCode_Null_Pointer	EnumErrorCode_Null_Pointer
DBRErrorCode_License_Invalid	EnumErrorCode_License_Invalid
DBRErrorCode_License_Expired	EnumErrorCode_License_Expired
DBRErrorCode_File_Not_Found	EnumErrorCode_File_Not_Found
DBRErrorCode_Filetype_Not_Supported	EnumErrorCode_Filetype_Not_Supported
DBRErrorCode_BPP_Not_Supported	EnumErrorCode_BPP_Not_Supported
DBRErrorCode_Index_Invalid	EnumErrorCode_Index_Invalid
DBRErrorCode_Barcode_Format_Invalid	EnumErrorCode_Barcode_Format_Invalid
DBRErrorCode_Custom_Region_Invalid	EnumErrorCode_Custom_Region_Invalid
DBRErrorCode_Max_Barcode_Number_Invalid	EnumErrorCode_Max_Barcode_Number_Invalid
DBRErrorCode_Image_Read_Failed	EnumErrorCode_Image_Read_Failed
DBRErrorCode_TIFF_Read_Failed	EnumErrorCode_TIFF_Read_Failed
DBRErrorCode_QR_License_Invalid	EnumErrorCode_QR_License_Invalid
DBRErrorCode_1D_Lincese_Invalid	EnumErrorCode_1D_Lincese_Invalid
DBRErrorCode_PDF417_License_Invalid	EnumErrorCode_PDF417_License_Invalid
DBRErrorCode_Datamatrix_License_Invalid	EnumErrorCode_Datamatrix_License_Invalid
DBRErrorCode_PDF_Read_Failed	EnumErrorCode_PDF_Read_Failed
DBRErrorCode_PDF_DLL_Missing	EnumErrorCode_PDF_DLL_Missing
DBRErrorCode_Page_Number_Invalid	EnumErrorCode_Page_Number_Invalid

DBRErrorCode_Custom_Size_Invalid	EnumErrorCode_Custom_Size_Invalid
DBRErrorCode_Custom_Modulesize_Invalid	EnumErrorCode_Custom_Modulesize_Invalid
DBRErrorCode_Recognition_Timeout	EnumErrorCode_Recognition_Timeout
DBRErrorCode_Json_Parse_Failed	EnumErrorCode_Json_Parse_Failed
DBRErrorCode_Json_Type_Invalid	EnumErrorCode_Json_Type_Invalid
DBRErrorCode_Json_Key_Invalid	EnumErrorCode_Json_Key_Invalid
DBRErrorCode_Json_Value_Invalid	EnumErrorCode_Json_Value_Invalid
DBRErrorCode_Json_Name_Key_Missing	EnumErrorCode_Json_Name_Key_Missing
DBRErrorCode_Json_Name_Value_Duplicated	EnumErrorCode_Json_Name_Value_Duplicated
DBRErrorCode_Template_Name_Invalid	EnumErrorCode_Template_Name_Invalid
DBRErrorCode_Json_Name_Reference_Invalid	EnumErrorCode_Json_Name_Reference_Invalid
DBRErrorCode_Parameter_Value_Invalid	EnumErrorCode_Parameter_Value_Invalid
DBRErrorCode_Domain_Not_Matched	EnumErrorCode_Domain_Not_Matched
DBRErrorCode_ReservedInfo_Not_Matched	EnumErrorCode_ReservedInfo_Not_Matched
DBRErrorCode_AZTEC_License_Invalid	EnumErrorCode_AZTEC_License_Invalid
DBRErrorCode_License_Dll_Missing	EnumErrorCode_License_Dll_Missing
DBRErrorCode_Licensekey_Not_Matched	EnumErrorCode_Licensekey_Not_Matched
DBRErrorCode_Licensefile_Invalide	EnumErrorCode_Licnse_Content_Invalid
DBRErrorCode_Requested_Failed	EnumErrorCode_Requested_Failed
DBRErrorCode_Licensefile_Expried	Removed

Other renamed or removed enumerations

Version 6.x	Version 7.x
EnumRegionPredetectionMode	
RegionPredetectionDisable	Removed
RegionPredetectionEnable	Removed
EnumTextFilterMode	
TextFilterDisable	Removed
TextFilterEnable	Removed