Dynamsoft

15 Years of Experience in TWAIN SDKs
and Version Control Solutions

Developer's Guide

version 6.4

Dynamsoft Barcode Reader

Table of Contents

Overview 1.1

Build a "Hello World" Application 1.2

- Windows Edition - C 1.2.1
- Windows Edition - C++ 1.2.2
- Windows Edition - C# 1.2.3
- Windows Edition - VB.NET 1.2.4
- Linux Edition - C 1.2.5
- Linux Edition - C++ 1.2.6
- Windows/Linux/Mac Edition - Java 1.2.7
- iOS Edition - Objective-C 1.2.8
- Android Edition - Java 1.2.9
- JavaScript Edition - JavaScript 1.2.10

Barcode Reading Settings 1.3

- Read Barcodes with Default Mode 1.3.1
- Specify Which Barcode Type to Read 1.3.2
- Read a Single Barcode 1.3.3
- Read from a Region 1.3.4
- Customize Barcode Scanning for Specific Use Case 1.3.5

Decoding Methods 1.4

- DecodeFile 1.4.1
- DecodeBase64String 1.4.2
- DecodeBitmap 1.4.3
- DecodeBuffer 1.4.4
- DecodeFileInMemory 1.4.5

Useful Resources 1.5

Overview

Dynamsoft's Barcode Reader SDK enables you to efficiently embed barcode reading functionality in your application using just a few lines of code. This can save you from months of added development time and extra costs.

With the Dynamsoft Barcode Reader SDK, you can decode barcodes from various image file formats (bmp, jpg, png, gif, tiff and pdf) as well as device-independent bitmap (DIB) which has just been obtained from cameras and scanners, etc.

The SDK has the following editions available:

Edition	Supported Programing Languages
Windows Edition	C, C++, C#, VB .NET, PHP, Java
Linux Edition	C, C++, Java
JavaScript Edition (Preview)	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

You can use the SDK in various development environments such as Visual Studio .NET (C# / VB.NET), Visual C++, Xcode, 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: QRCode, PDF417, DataMatrix, Aztec

Build a "Hello World" Application

To get started, we'll first show how to build a "Hello World" application that reads barcodes from a static image using the following programming languages:

- Windows Edition - C
- Windows Edition - C++
- Windows Edition - C#
- Windows Edition - VB.NET
- Linux Edition - C
- Linux Edition - C++
- Windows/Linux/Mac Edition - Java
- iOS Edition - Objective-C
- Android Edition - Java
- JavaScript Edition (Preview) - JavaScript

Windows Edition - C

This article shows how to create a C version of the traditional "Hello World" program in a desktop application using Dynamsoft Barcode Reader Windows Edition.

Steps

1. Start Visual Studio and create a new Win32 Console Application in C. Let's name it BarcodeReadDemo_C
2. Add headers and libs in `BarcodeReadDemo_C.c`.

   ```
   #include <stdio.h>
   #include "<relative path>/DynamsoftBarcodeReader.h"

   #ifdef _WIN64
   #pragma comment(lib, "<relative path>/DBRx64.lib")
   #else
   #pragma comment(lib, "<relative path>/DBRx86.lib")
   #endif
   ```

   Note: Please replace `<relative path>` in the code with the relative path to the `BarcodeReadDemo_C.c` file. Typically, The `DynamsoftBarcodeReader.h` file can be found in "[INSTALL FOLDER]\Components\C_C++\Include\", and the .lib files can be found in "[INSTALL FOLDER]\Components\C_C++\Lib\".

3. In `BarcodeReadDemo_C.c`, please update the main function.

Please update the `<your image file full path>` and `<your license key here>` with valid values respectively in the code.

If you already downloaded and installed Dynamsoft Barcode Reader 30-day free trial, you can find the license in the "C:\Program Files (x86)\Dynamsoft\Barcode Reader {version number}\LicenseManager.exe". If not, please download it here.

```
int main()
{
 // Define variables
 void *hBarcode = NULL;
 int iRet = -1;
 int iIndex = 0;
 STextResultArray *pResult = NULL;

 hBarcode = DBR_CreateInstance();

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

 // 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->nResultsCount);
 for (iIndex = 0; iIndex < pResult->nResultsCount; iIndex++)
 {
     printf("Result %d\n", iIndex + 1);
     printf("Barcode Format: %s\n", pResult->ppResults[iIndex]->pszBarcodeFormatStr
ing);
     printf("Text reads: %s \n", pResult->ppResults[iIndex]->pszBarcodeText);
 }

 // Finally release BarcodeResultArray and destroy instance
 DBR_FreeTextResults(&pResult);
 DBR_DestroyInstance(hBarcode);

 return 0;
```

```
}
```

4. Run the project.

   Build the application, and copy the `x86\DynamsoftBarcodeReaderx86.dll`, `x86\vcomp110.dll`, `x86\DynamicPdf.dll` in x86 mode (or `x64\DynamsoftBarcodeReaderx64.dll`, `x64\vcomp110.dll`, `x64\DynamicPdfx64.dll` in x64 mode) to the .EXE folder. The DLLs can be found in "[INSTALL FOLDER]\Components\C_C++\Redist\".

   To verify it is working, please open the cmd.exe, and execute the .EXE with a barcode image.

   To deploy your application, please make sure the DLLs are in the same folder as the EXE file. If your barcodes are from PDF files, `DynamicPdf.dll` and `DynamicPdfx64.dll` are mandatory.

Windows Edition - C++

This guide shows how to create a C++ version of the traditional "Hello World" program in a desktop application using Dynamsoft Barcode Reader Windows Edition.

Steps

1. Start Visual Studio and create a new Win32 Console Application in C++. Let's name it `BarcodeReadDemo_CPP`.

2. Add headers and libs in `BarcodeReadDemo_CPP.cpp`.

```
#include <stdio.h>
#include "<relative path>/DynamsoftBarcodeReader.h"

#ifdef _WIN64
#pragma comment(lib, "<relative path>/DBRx64.lib")
#else
#pragma comment(lib, "<relative path>/DBRx86.lib")
#endif
```

   Note Please replace `<relative path>` in the code with the relative path to the `BarcodeReadDemo_CPP.cpp file`. Typically, The `DynamsoftBarcodeReader.h` file can be found in "[INSTALL FOLDER]\Components\C_C++\Include\", and the .lib files can be found in "[INSTALL FOLDER]\Components\C_C++\Lib\".

3. In the `BarcodeReadDemo_CPP.cpp`, please update the main function.

Please update the `<your image file full path>` and `<your license key here>` with valid values respectively in the code.

If you already downloaded and installed Dynamsoft Barcode Reader 30-day free trial, you can find the license in the "C:\Program Files (x86)\Dynamsoft\Barcode Reader {version number}\LicenseManager.exe". If not, please download it here.

```
int main()
{
// Define variables
int iRet = -1;
STextResultArray *paryResult = NULL;

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

// 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->nResultsCount);
for (int iIndex = 0; iIndex < paryResult->nResultsCount; iIndex++)
{
    printf("Result %d\r\n", iIndex + 1);
    printf("BarcodeFormat: %s\r\n", paryResult->ppResults[iIndex]->pszBarcodeForma
tString);
    printf("Text read: %s\r\n", paryResult->ppResults[iIndex]->pszBarcodeText);
}

// Finally release BarcodeResultArray
CBarcodeReader::FreeTextResults(&paryResult);

return 0;
}
```

4. Run the project

Build the application, and copy the `x86\DynamsoftBarcodeReaderx86.dll`, `x86\vcomp110.dll`, `x86\DynamicPdf.dll` in x86 mode (or `x64\DynamsoftBarcodeReaderx64.dll`, `x64\vcomp110.dll`, `x64\DynamicPdfx64.dll` in x64 mode) to the .EXE folder. The DLLs can be found in "[INSTALL FOLDER]\Components\C_C++\Redist\".

To verify it is working, please open the cmd.exe, and execute the .EXE with a barcode image.

To deploy your application, please make sure the DLLs are in the same folder as the EXE file. If your barcodes are from PDF files, `DynamicPdf.dll` and `DynamicPdfx64.dll` are mandatory.

Windows Edition - C#

This guide shows how to create a C# version of the traditional "Hello World" program in a desktop application.

Steps

1. Start Visual Studio and create a new Windows Forms Application in C#.
2. Right click **References** in the Solution Explorer to add `Dynamsoft.BarcodeReader.dll` to the project.

   Note: The DLLs can be found in the installation folder. Typically, they are under:

   - **For .NET Framework 2.0-3.5:**

     "[INSTALL FOLDER]\Components\DotNet\2.0\"

   - **For .NET Framework 4.0 and later:**

     "[INSTALL FOLDER]\Components\DotNet\4.0\"
3. Add namespace:

   ```
   using Dynamsoft.Barcode;
   ```
4. Create a new button on the form and replace the contents of button1_Click with the following code.

   Note: Please update the `"<your image file full path>"` and `"<your license key here>"` with valid values respectively in the code.

   If you already downloaded and installed Dynamsoft Barcode Reader 30-day free trial, you can find the license in the "C:\Program Files (x86)\Dynamsoft\Barcode Reader {version number}\LicenseManager.exe". If not, please download it here.

```
 private void button1_Click(object sender, EventArgs e)
 {
     try
     {
         BarcodeReader dbr = new BarcodeReader();
         dbr.LicenseKeys = "<your license key here>";
         TextResult[] aryResult = dbr.DecodeFile("<your image file full path>", "")
; // leave the template name empty ("") will use  default settings for recognition

         if (aryResult.Length == 0)
         {
             Console.WriteLine("No barcode found.");
         }
         else
         {
             for (var i = 0; i < aryResult.Length; i++)
             {
                 Console.WriteLine("Barcode: {0}", (i + 1));
                 Console.WriteLine("BarcodeFormat: {0}", aryResult[i].BarcodeFormat
.ToString());
                 Console.WriteLine("BarcodeText: {0}", aryResult[i].BarcodeText);
             }
         }
     }
     catch (BarcodeReaderException exp)
     {
         Console.WriteLine("Error: {0}, {1}", exp.Code.ToString(), exp.Message);
     }
 }
```

5. Run the project.

   To deploy your application, please make sure the DLL is in the same folder as the EXE file.

Windows Edition - VB.NET

This guide shows how to create a VB.NET version of the traditional "Hello World" program in a desktop application.

Steps

1. Start Visual Studio and create a new Windows Forms Application in VB.NET.
2. Right click **References** in the Solution Explorer to add `Dynamsoft.BarcodeReader.dll` to the project.

Note: The DLLs can be found in the installation folder. Typically, they are under:

- **For .NET Framework 2.0-3.5:**

  "[INSTALLATION FOLDER]\Components\DotNet\2.0\"

- **For .NET Framework 4.0 and later:**

  "[INSTALLATION FOLDER]\Components\DotNet\4.0\"

3. Add namespace:

   ```
   Imports Dynamsoft.Barcode
   ```

4. Create a new button on the form and replace the contents of button1_Click with the following code.

   Note: Please update the `"<your image file full path>"` and `"<your license key here>"` with valid values respectively in the code.

   If you already downloaded and installed Dynamsoft Barcode Reader 30-day free trial, you can find the license in the "C:\Program Files (x86)\Dynamsoft\Barcode Reader {version number}\LicenseManager.exe". If not, please download it here.

```
 Private Sub Button1_Click(sender As Object, e As EventArgs) Handles Button1.Click
     Dim reader As New BarcodeReader
     reader.LicenseKeys = "<Put your license key here>"
     Dim aryResult As TextResult()
     Try
         aryResult = reader.DecodeFile("<your image file full path>", "") ' leave t
he template name empty ("") will use  default settings for recognition
         If aryResult.length == 0 Then
             MessageBox.Show("No barcode found.")
             Return
         End If

         Dim strInfo As String
         strInfo = "Total barcode(s) found: " + aryResult.Length.ToString() + vbCrL
f
         For i = 0 To aryResult.Length - 1
             strInfo += "Barcode: " + (i + 1).ToString() + ":" + vbCrLf
             strInfo += "BarcodeFormat: " + aryResult(i).BarcodeFormat.ToString() +
 vbCrLf
             strInfo += "BarcodeText: " + aryResult(i).BarcodeText.ToString() + vbC
rLf + vbCrLf
         Next
         MessageBox.Show(strInfo)
     Catch ex As BarcodeReaderException
         MessageBox.Show("Error Code:" + ex.Code.ToString() + vbCrLf + "Error Strin
g: " + ex.Message)
```

```
        End Try
    End Sub
```

5. Run the project.

   To deploy your application, please make sure the DLL is in the same folder as the EXE file.

Linux Edition - C

This article shows how to create a C version of the traditional "Hello World" program in a desktop application using Dynamsoft Barcode Reader Linux Edition.

Steps

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 `<relative path>` with your own relative path in the code.

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

```
#include <stdio.h>
```

```c
#include "<relative path>/DynamsoftBarcodeReader.h"

int main()
{
    // Define variables
    void *hBarcode = NULL;
    int iRet = -1;
    int iIndex = 0;
    STextResultArray *pResult = NULL;

    hBarcode = DBR_CreateInstance();

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

    // 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(iR
et));
        return iRet;
    }

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

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

Note: Please update the `<relative path>`, `<your image file full path>` and `<your license key here>` with valid values respectively in the code.

If you don't have a trial license or it has expired, you can request a new trial license here.

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

Linux Edition - C++

This article shows how to create a C++ version of the traditional "Hello World" program in a desktop application using Dynamsoft Barcode Reader Linux Edition.

Steps

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 `<relative path>` with your own relative path in the code.

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

```
#include <stdio.h>
#include "<relative path>/DynamsoftBarcodeReader.h"
```

```
int main()
{
// Define variables
int iRet = -1;
STextResultArray *paryResult = NULL;

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

// 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->nResultsCount);
for (int iIndex = 0; iIndex < paryResult->nResultsCount; iIndex++)
{
    printf("Result %d\r\n", iIndex + 1);
    printf("BarcodeFormat: %s\r\n", paryResult->ppResults[iIndex]->pszBarcodeForma
tString);
    printf("Text read: %s\r\n", paryResult->ppResults[iIndex]->pszBarcodeText);
}

// Finally release BarcodeResultArray
CBarcodeReader::FreeTextResults(&paryResult);

return 0;
}
```

Note Please update the `<relative path>`, `<your image file full path>` and `<your license key here>` with valid values respectively in the code.

If you don't have a trial license or it has expired, you can request a new trial license here.

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

Mobile Edition - iOS

This following step-by-step guide provides information on how to create your first native application for iOS using our Dynamsoft Barcode Reader SDK with iOS.

1. Download the Dynamsoft Barcode Reader library

   If you haven't downloaded the Dynamsoft Barcode Reader 30-day free trial, please download it here.

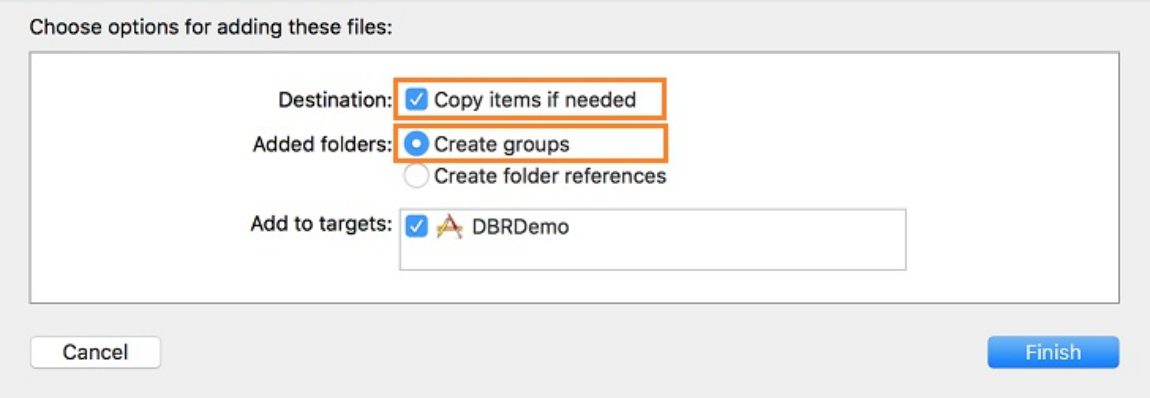
   Note: You may need to sign in to download the trial.

2. Create a new iOS project in Xcode

3. Add the barcode reader framework to your project

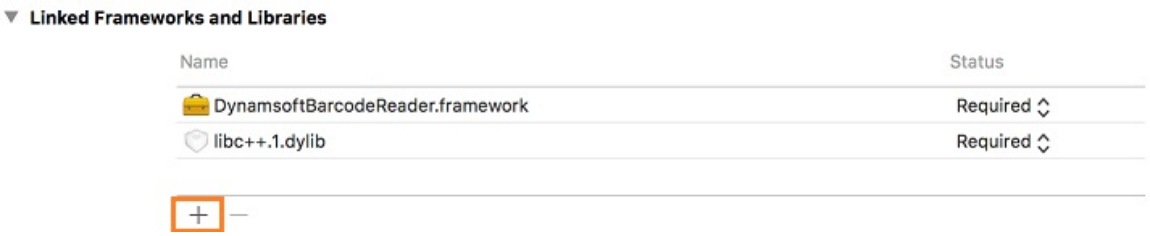
   3.1 Unzip the downloaded ZIP file `dbr-ios-{version number}.zip` and open the folder named `SDK`, it contains the DynamsoftBarcodeReader framework.

   3.2 Drag and drop the framework into your Xcode project. Make sure to select **Create groups** and **Copy items if needed** to copy the framework into your project's folder.

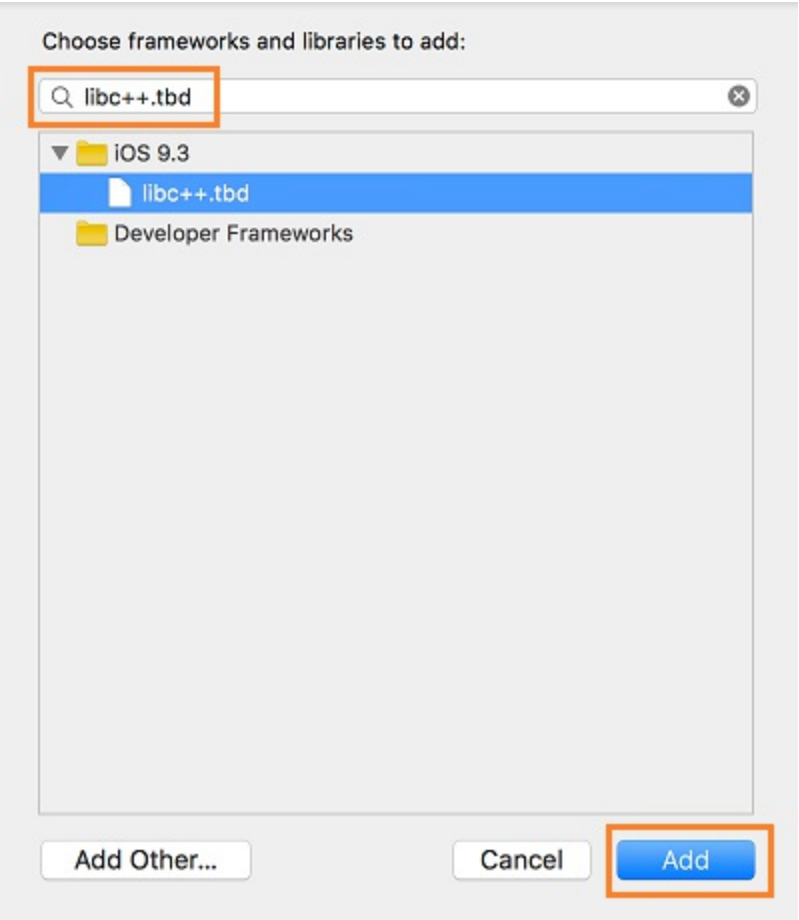


4. Add the required .tbd/.dylib file to your project

   4.1 Go to the **General** tab of your Xcode project, under **Linked Frameworks and Libraries** section, please click **+** button.

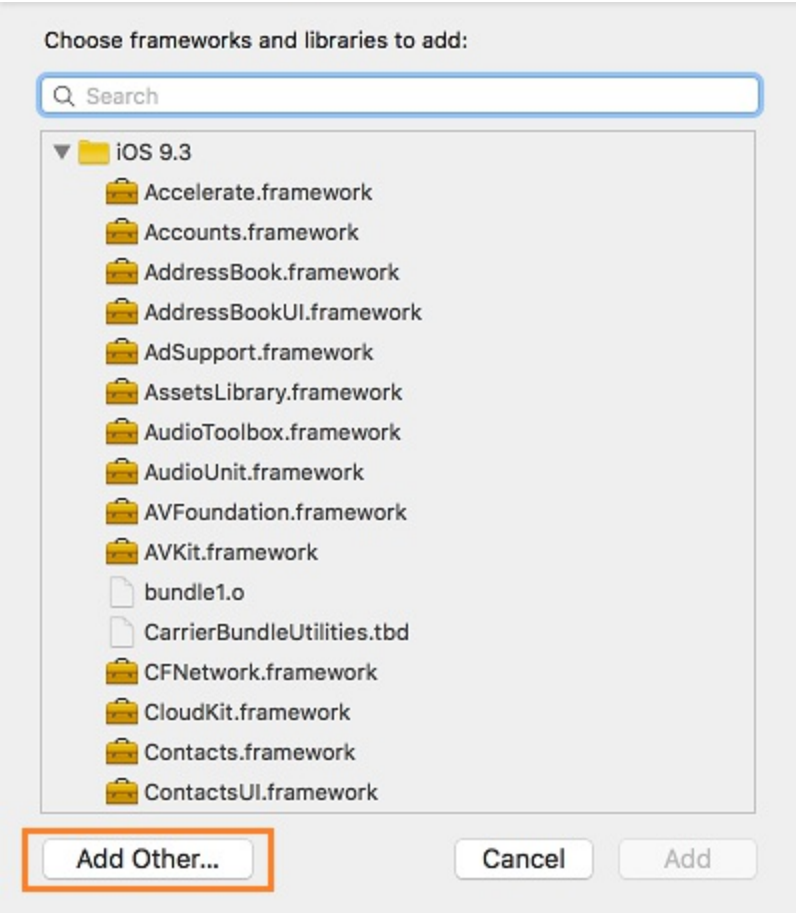


   4.2 Search for `libc++.tbd` file, select it and click **Add** button. Then the `libc++.tbd` file will be copied to your project.

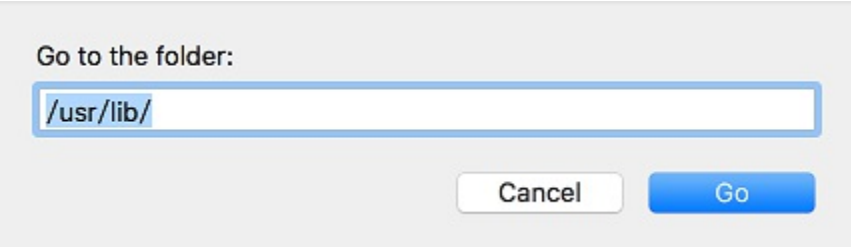


Note: For Xcode version 7 and earlier, please include `libc++.dylib` instead of `libc++.tbd` into your project. Here are the detailed steps:

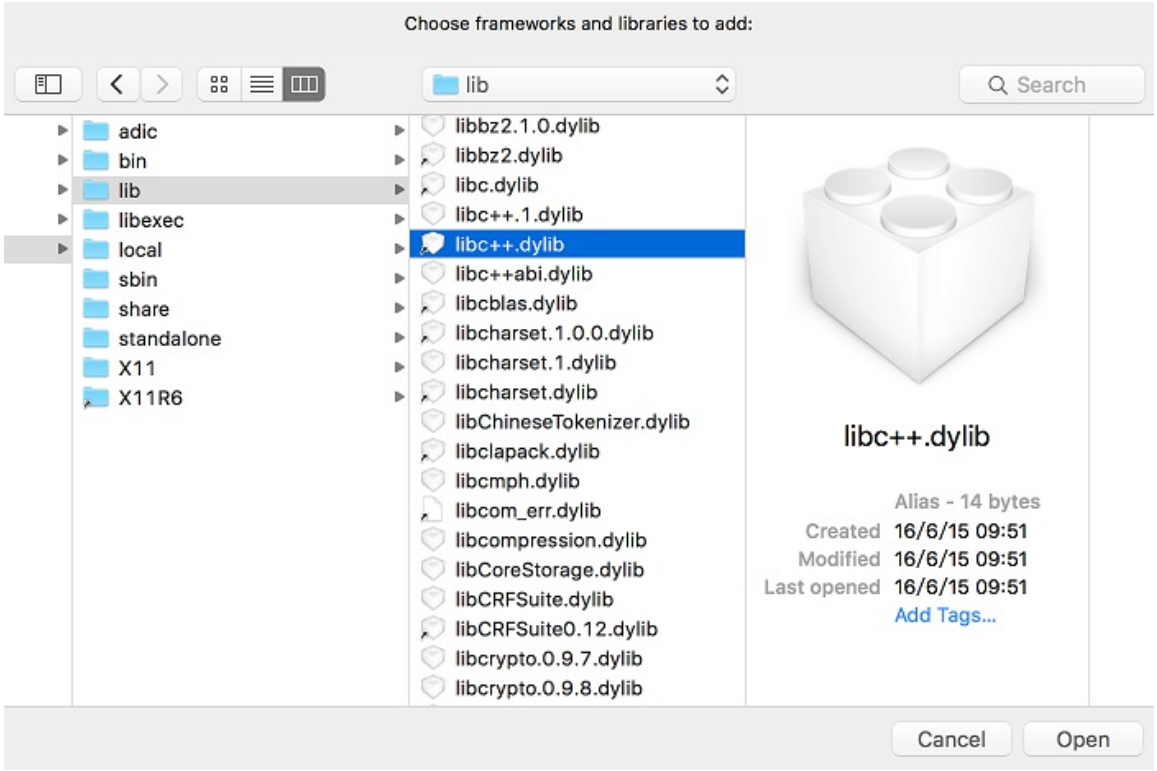
In **General** -> **Linked Frameworks and Libraries** -> click **+** button -> **Add Other...**



Once in the file selection window, press **"CMD+Shift+G"** (Go to folder) and type /usr/lib/



From /user/lib you can find the libc++.dylib file, select it and click **Open**



5. Import the framework's header

```
#import <DynamsoftBarcodeSDK/DynamsoftBarcodeSDK.h>
```

6. Add code to start barcode recognition process

   The basic project has been set up, you can now begin developing your own application. The following code demonstrates initializing the DynamsoftBarcodeReader and starting the recognition process:

```
#import "ViewController.h"
#import <DynamsoftBarcodeSDK/DynamsoftBarcodeSDK.h>

@interface ViewController ()
@end
@implementation ViewController

- (void)viewDidLoad {
 [super viewDidLoad];
 // Do any additional setup after loading the view, typically from a nib.

 // Create and initialize a DynamsoftBarcodeReader.
 DynamsoftBarcodeReader *dbr;
 // Please replace "0C859***624A5" with your own license.
 // Note: If you do not have a valid license for the SDK, some characters of the ba
rcode results
 // will be replaced with "***".
```

```
    dbr = [[DynamsoftBarcodeReader alloc] initWithLicense:@"0C859***624A5"];

    UIImage *image =[UIImage imageNamed:@"BarcodeImage.png"];

    NSError* error = nil;
    NSArray* readResult = [dbr decodeImage:image withTemplate:@"" error:&error];
    if (error)
    {
        NSLog(@"%@", error);
    }
    else
    {
        if (readResult.count > 0)
        {
            TextResult* result = readResult[0];
            NSLog(@"%@: %@", image, result.barcodeText);
        }
        else
        {
            NSLog(@"%@: no barcode found.",image);
        }
    }
}

 - (void)didReceiveMemoryWarning {
     [super didReceiveMemoryWarning];
     // Dispose of any resources that can be recreated.
 }

@end
```

Windows/Linux/Mac Edition - Java

This guide shows how to create a Java version of the traditional "Hello World" program in a desktop application.

Steps

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{version number}.jar** and click **Apply**.
3. Import the header

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

4. Replace the contents of **HelloDBR** with the following code.

   Note Please update the `<your image file full path>` and `<your license key here>` with valid values respectively in the code.

   If you don't have a trial license or it has expired, you can request a new trial license here.

```
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.

Mobile Edition - Android

This following step-by-step guide provides information on how to integrate the Dynamsoft Barcode Reader SDK into your Android app.

1. Download the Dynamsoft Barcode Reader SDK

   If you haven't downloaded the Dynamsoft Barcode Reader 30-day free trial, please download it here.

   Note: You may need to sign in to download the trial.

2. Create a new Empty Activity project in Android Studio
3. Import .aar package and add dependency

3.1 Right click app -> select **New** -> **Module** -> Import .JAR/.AAR Package -> select `DynamsoftBarcodeReader.aar`

3.2 Open `build.gradle` file and add the following code:

```
repositories {
 flatDir {
     dirs 'libs'
 }
}
```

Then add .aar reference in the dependencies{}, like this:

Change from

```
implementation fileTree(dir: 'libs', include: ['*.jar'])
```

to

```
implementation fileTree(dir: 'libs', include: ['*.jar', '*.aar'])
```

4. Add storage access permission

   This sample will need to access `QRcode.png` from your storage card, so please add the following QRCode image to your device.

   To add storage access permission, please open `AndroidManifest.xml` file, put a " element in your app manifest.

```
<uses-permission android:name="android.permission.READ_EXTERNAL_STORAGE"/>
```

5. Import the headers

```
import com.dynamsoft.barcode.BarcodeReader;
import com.dynamsoft.barcode.BarcodeReaderException;
import com.dynamsoft.barcode.TextResult;
```

6. Edit APP user interface

   Open `activity_main.xml`, switch to the design view, add a button and a label, like this:

7. Add the code

In the `MainActivity.java`:

```
package com.example.leo.demo;
import android.Manifest;
import android.content.pm.PackageManager;
import android.os.Environment;
import android.support.annotation.NonNull;
import android.support.v4.app.ActivityCompat;
import android.support.v4.content.ContextCompat;
import android.support.v7.app.AppCompatActivity;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.TextView;
import android.widget.Toast;

import com.dynamsoft.barcode.BarcodeReader;
import com.dynamsoft.barcode.BarcodeReaderException;
import com.dynamsoft.barcode.TextResult;

public class MainActivity extends AppCompatActivity {

    private Button btnRead;
    private TextView textV;
    BarcodeReader barcodeReader;
    TextResult[] results;
    String filePath;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        requestPermission();
        btnRead = findViewById(R.id.btnRead);
        textV = findViewById(R.id.textView);
        try{
            barcodeReader = new BarcodeReader("");
            filePath = Environment.getExternalStorageDirectory().getCanonicalPath(
) + "/" + "QRcode.png";
            eventHandle();
        }catch(BarcodeReaderException e){
            e.printStackTrace();
        }catch (Exception e){
            e.printStackTrace();
        }
    }

    public void eventHandle(){
```

```
        btnRead.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                try{
                    results = barcodeReader.decodeFile(filePath,"");
                    if(results!=null&&results.length>0)
                        textV.setText(results[0].barcodeText);
                    else textV.setText("The file: \""+ filePath+"\" not found!");
                }catch (BarcodeReaderException e){
                    e.printStackTrace();
                }catch (Exception e){
                    e.printStackTrace();
                }
            }
        });
    }

    private void requestPermission(){
 if (Build.VERSION.SDK_INT>22){
 try{
        if(ContextCompat.checkSelfPermission(this, Manifest.permission.READ_EXTERN
AL_STORAGE)!= PackageManager.PERMISSION_GRANTED){
            ActivityCompat.requestPermissions(this,new String[]{Manifest.permissio
n.READ_EXTERNAL_STORAGE},1);
        }
    }catch(Exception e){
    e.printStackTrace();
    }
    }
    }

    @Override
    public void onRequestPermissionsResult(int requestCode, @NonNull String[] perm
issions, @NonNull int[] grantResults) {
        super.onRequestPermissionsResult(requestCode, permissions, grantResults);
        switch (requestCode){
            case 1:
                if(grantResults.length<=0&&grantResults[0]!=PackageManager.PERMISS
ION_GRANTED) {
                    Toast.makeText(this, "Permission denied",Toast.LENGTH_SHORT).s
how();
                }
        }
    }
 }
```

In the `AndroidManifest.xml` :

```
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
 package="com.example.leo.demo">
 <uses-permission android:name="android.permission.READ_EXTERNAL_STORAGE"/>

 <application
     android:allowBackup="true"
     android:icon="@mipmap/ic_launcher"
     android:label="@string/app_name"
     android:roundIcon="@mipmap/ic_launcher_round"
     android:supportsRtl="true"
     android:theme="@style/AppTheme">
     <activity android:name=".MainActivity">
         <intent-filter>
             <action android:name="android.intent.action.MAIN" />

             <category android:name="android.intent.category.LAUNCHER" />
         </intent-filter>
     </activity>
 </application>
</manifest>
```

In the `build.gradle` (Module:app):

```
apply plugin: 'com.android.application'

android {
 compileSdkVersion 27
 defaultConfig {
     applicationId "com.example.leo.demo"
     minSdkVersion 15
     targetSdkVersion 27
     versionCode 1
     versionName "1.0"
     testInstrumentationRunner "android.support.test.runner.AndroidJUnitRunner"
 }
 buildTypes {
     release {
         minifyEnabled false
         proguardFiles getDefaultProguardFile('proguard-android.txt'), 'proguard-ru
les.pro'
     }
 }
 repositories {
     flatDir {
         dirs 'libs'
     }
 }
```

```
}

dependencies {
 implementation fileTree(dir: 'libs', include: ['*.jar', '*.aar'])
 implementation 'com.android.support:appcompat-v7:27.1.1'
 implementation 'com.android.support.constraint:constraint-layout:1.1.2'
 testImplementation 'junit:junit:4.12'
 androidTestImplementation 'com.android.support.test:runner:1.0.2'
 androidTestImplementation 'com.android.support.test.espresso:espresso-core:3.0.2'
}
```

8. Build and run the APP

   When you run the application, you will be asked for the permission to access your device, please grant the access.

JavaScript Edition - JavaScript

Dynamsoft Barcode Reader JavaScript Edition is a JavaScript API for barcode scanning based on the WebAssembly technology. The library is capable of scanning barcodes from a static image and from a live video stream directly. It also supports reading multiple barcodes at a time.

Please refer to this page for more info on how to build a "HelloWorld" application.

Barcode Reading Settings

- Read Barcodes with Default Mode
- Specify Which Barcode Type to Read
- Read a Single Barcode
- Read from a Specified Region
- Customize Barcode Scanning for Specific Use Case

Read Barcodes with Default Mode

The general steps to use the SDK are as follows:

1. Initialize the SDK with a license key. Typically, it only needs to be done once in the application, before calling decoding methods.
2. Set optional barcode reading settings, such as barcode type, scanning region, etc. For new customers, you can skip this step and it will use the default settings to search for all barcode types at any angle in the whole image. For customers who want to customize the barcode reading settings, please refer to the following chapters.
3. Perform barcode reading.

Below is the code snippet for using default settings in C#.

```
    BarcodeReader barcodeReader = new BarcodeReader();
    // Replace <Put your license key here> with your own license.
    barcodeReader.LicenseKeys = "<Put your license key here>";
    TextResult[] aryResult = barcodeReader.DecodeFile(@"C:\Program Files (x86)\Dynamsoft\Barcode Reader 6.4\Images\AllSupportedBarcodeTypes.tif", string.Empty);
```

If you have already installed the Dynamsoft Barcode Reader 30-day free trial, you can find the license in the "C:\Program Files (x86)\Dynamsoft\Barcode Reader {version number}\LicenseManager.exe".

Note that if you don't pass a license to the SDK or the license has expired, the SDK will continue to function normally but the last three characters of the barcode result will be masked with "*".

Below is a list of the default values of all public runtime barcode reading settings:

Parameter	Default Value
Name	N/A
AntiDamageLevel	9
BarcodeFormatIds	503317503 (All barcode formats)

ColourImageConvertMode	CICM_Auto
DeblurLevel	9
EnableFillBinaryVacancy	true
ExpectedBarcodesCount	0
GrayEqualizationSensitivity	0
LocalizationAlgorithmPriority	"" (empty string - The library will automatically use optimized localization algorithm first)
MaxAlgorithmThreadCount	4
MaxBarcodesCount	0x7fffffff
PDFRasterDPI	300
RegionPredetectionMode	RPM_Disable
ScaleDownThreshold	2300
TextFilterMode	TFM_Enable
TextureDetectionSensitivity	5
Timeout	10000

Specify Which Barcode Type to Read

By default, the SDK will read the following supported barcode types from the image:

- 1D barcodes: Code39, Code93, Code128, Codabar, ITF, EAN13, EAN8, UPCA, UPCE, INDUSTRIAL 2 OF 5;
- 2D barcodes: QRCode, PDF417, DataMatrix, Aztec (supported since version 6.3).

If your full license only covers one or part of barcode types, you must specify which barcode type(s) the SDK should read. For example, to enable only 1D barcode reading, please use the following:

```
BarcodeReader barcodeReader = new BarcodeReader();
barcodeReader.LicenseKeys = "Put your license key here.";
// get the runtime settings from the BarcodeReader object
PublicRuntimeSettings settings = barcodeReader.GetRuntimeSettings();
//only have the barcode types which you have a license for included.
settings.mBarcodeFormatIds = 1023; // OneD barcode
// update the runtime settings to the BarcodeReader object
barcodeReader.UpdateRuntimeSettings(settings);
// setting the template name to empty will use the runtime settings you just updated


TextResult[] aryResult = barcodeReader.DecodeFile(@"C:\Program Files (x86)\Dynamsof
```

```
t\Barcode Reader 6.4\Images\AllSupportedBarcodeTypes.tif", string.Empty);
```

Read a Single Barcode

By default, the SDK will read as many barcodes as it can. You can specify the maximum number of barcodes to read before decoding according to your use case. This can increase the recognition efficiency.

```
    // get the runtime settings from the BarcodeReader object
    PublicRuntimeSettings settings = barcodeReader.GetRuntimeSettings();
    // limit it to decode only 1 barcode
    settings.mMaxBarcodesCount = 1;
    // update the runtime settings to the BarcodeReader object
    barcodeReader.UpdateRuntimeSettings(settings);
    // setting the template name to empty will use the runtime settings you just updated

    TextResult[] aryResult = barcodeReader.DecodeFile(@"C:\Program Files (x86)\Dynamsof
t\Barcode Reader 6.4\Images\AllSupportedBarcodeTypes.tif", string.Empty);
```

Read from a Region

By default, the barcode reader will search the whole image. 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 create a template string/file and define the area in the RegionDefinition. For more details on RegionDefinition, please refer to this chapter.

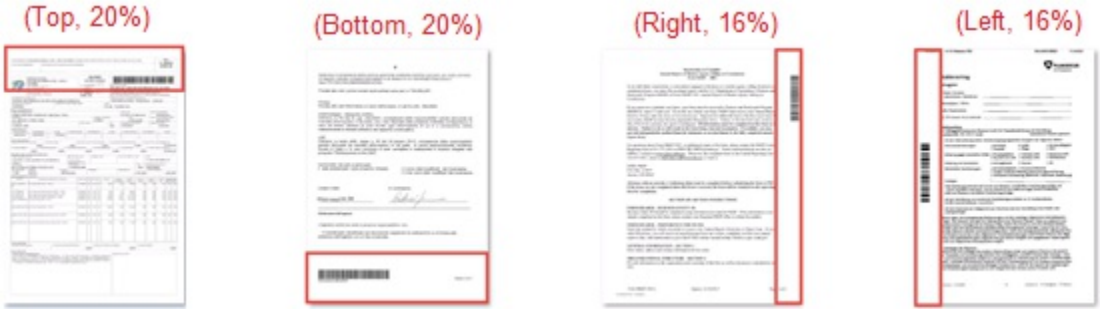
The following function shows how to create a template string and define the region:

```
    public TextResult[] DecodeSpecificRegionBarcodes(string strBarcodeFilePath, int iLe
ft, int iTop, int iRight, int iBottom)
    {
        BarcodeReader barcodeReader = new BarcodeReader();
        barcodeReader.LicenseKeys = "Put your license key here.";
        try
        {
            //create the BarcodeReader runtime setttings template string for the specif
ic region decode.
            string strTemplateName = "TemplateName";
            string strRegionName = "RegionName";
            //create a  template json string
            string tempTemplateJsonWithRegion = "{\"ImageParameter\": {\"Name\": \"" +
```

```
strTemplateName + "\",\"RegionDefinitionNameArray\": [\"" + strRegionName + "\"]},\"Reg
ionDefinitionArray\": [{\"Name\": \"" + strRegionName + "\",\"MeasuredByPercentage\": t
rue" + ",\"Left\":" + iLeft.ToString() + ",\"Top\":" + iTop.ToString() + ",\"Right\":"
+ iRight.ToString() + ",\"Bottom\":" + iBottom.ToString() + "}]}";

            // append the runtime settting template to the BarcodeReader object
            String strMsg = "";
            barcodeReader.AppendTplStringToRuntimeSettings(tempTemplateJsonWithRegion,
EnumConflictMode.ECM_Overwrite, out strMsg);
            // docode the barcode file
            return barcodeReader.DecodeFile(strBarcodeFilePath, strTemplateName);
        }
        catch (BarcodeReaderException exception)
        {
            System.Console.WriteLine(exception.Message);
            Trace.WriteLine(exception.Message);
        }
        return null;
    }
```

For example:




```
// top 20%:
DecodeSpecificRegionBarcodes(@"C:\Images\Image1.jpg", 0, 0, 100, 20);

// bottom 20%:
DecodeSpecificRegionBarcodes(@"C:\Images\Image2.jpg", 0, 80, 100, 100);

// right 16%:
DecodeSpecificRegionBarcodes(@"C:\Images\Image3.jpg", 84, 0, 100, 100);

// left 16%:
DecodeSpecificRegionBarcodes(@"C:\Images\Image4.jpg", 0, 0, 16, 100);
```

Customize Barcode Scanning for Specific Use Case

Dynamsoft Barcode Reader provides flexible APIs for you to customize the scanning settings for different usage scenarios. You can create barcode reading templates with settings such as: barcode region, barcode count, barcode format, deblur level, anti-image level, for best performance of your specific use case.

For more information, please refer to the Online Documentation.

Decoding Methods

In addition to reading barcodes from a static image the SDK also supports reading barcodes from a video stream, file in memory, base64 string, bitmap, etc. Here is a list of all decoding methods and related code snippet/samples:

Method	Description
DecodeFile	Reads barcodes from a specified image file.
DecodeBase64String	Reads barcodes from the base64 encoded string of a file.
DecodeBitmap	Reads barcodes from a bitmap. It will only decode the current page when handling multi-page images.
DecodeBuffer	Reads barcodes from raw buffer.
DecodeFileInMemory	Decodes barcodes from an image file in memory.

Code snippets are provided in C# as examples, but not all supported programming languages are covered in this guide. If you are looking for samples in other programming languages, please refer to our Code Gallery.

DecodeFile

We have talked about how to read barcodes from a specified file in the previous chapter. Please refer to here for more information on how to use the method.

DecodeBase64String

Reads barcodes from the base64 encoded string of a file.

Example:

```
    //convert the bitmap content to a base64 string
    Bitmap barCodeBitmap = new Bitmap(@"C:\Program Files (x86)\Dynamsoft\Barcode Reader
6.4\Images\AllSupportedBarcodeTypes.tif");
    MemoryStream ms = new MemoryStream();
    barCodeBitmap.Save(ms, System.Drawing.Imaging.ImageFormat.Jpeg);
    byte[] bmpBytes = new byte[ms.Length];
    ms.Position = 0;
    ms.Read(bmpBytes, 0, (int)ms.Length);
    ms.Close();
    string barcodeBase64String = Convert.ToBase64String(bmpBytes);

    //Decode the base64 string barcode
    TextResult[] aryResult = barcodeReader.DecodeBase64String(barcodeBase64String, stri
```

```
ng.Empty);
```

DecodeBitmap

Reads barcodes from a bitmap. It will only decode the current page when handling multi-page images.

Example:

```
    //create a bitmap from a file
    Bitmap barCodeBitmap = new Bitmap(@"C:\Program Files (x86)\Dynamsoft\Barcode Reader
 6.4\Images\AllSupportedBarcodeTypes.tif");
    //Decode the bitmap barcode
    TextResult[] aryResult = barcodeReader.DecodeBitmap(barCodeBitmap, string.Empty);
```

DecodeBuffer

Reads barcodes from raw buffer.

Example:

```
    //create the image raw buffer from the barcode file
    Bitmap barCodeBitmap = new Bitmap(strBarcodeFilePath);
    EnumImagePixelFormat emImagePixelFormat = EnumImagePixelFormat.IPF_RGB_888;
    switch (barCodeBitmap.PixelFormat)
    {
        case PixelFormat.Format32bppArgb:
        case PixelFormat.Format32bppPArgb:
        case PixelFormat.Format32bppRgb:
            emImagePixelFormat = EnumImagePixelFormat.IPF_ARGB_8888;
            break;
        case PixelFormat.Format24bppRgb:
            emImagePixelFormat = EnumImagePixelFormat.IPF_RGB_888;
            break;
        case PixelFormat.Format1bppIndexed:
            emImagePixelFormat = EnumImagePixelFormat.IPF_Binary;
            break;
        case PixelFormat.Format8bppIndexed:
            emImagePixelFormat = EnumImagePixelFormat.IPF_GrayScaled;
            break;
    }
    Rectangle rect = new Rectangle(0, 0, barCodeBitmap.Width, barCodeBitmap.Height);
    BitmapData barCodeBitmapData = barCodeBitmap.LockBits(rect, System.Drawing.Imaging.
ImageLockMode.ReadWrite, barCodeBitmap.PixelFormat);
        // Declare an array to hold the bytes of the bitmap.
    int imageBytesSize = Math.Abs(barCodeBitmapData.Stride) * barCodeBitmap.Height;
```

```
    byte[] barcodeImageRawBuffer = new byte[imageBytesSize];
        // Get the address of the first line.
    IntPtr ptr = barCodeBitmapData.Scan0;
        // Copy the RGB values into the byte array.
    System.Runtime.InteropServices.Marshal.Copy(ptr, barcodeImageRawBuffer, 0, imageBytesSize);
    barCodeBitmap.UnlockBits(barCodeBitmapData);

    int iBarcodeImageWidth = barCodeBitmap.Width;
    int iBarcodeImageHeight = barCodeBitmap.Height;
    int iBarcodeImageStride = barCodeBitmapData.Stride;
    //Reads barcodes from raw buffer.
    TextResult[] aryResult = barcodeReader.DecodeBuffer(barcodeImageRawBuffer, iBarcodeImageWidth, iBarcodeImageHeight, iBarcodeImageStride, emImagePixelFormat, string.Empty);
```

DecodeFileInMemory

Decodes barcodes from an image file in memory.

Example:

```
    // read the file content into byte array
    FileStream fileStream = new FileStream(@"C:\Program Files (x86)\Dynamsoft\Barcode Reader 6.4\Images\AllSupportedBarcodeTypes.tif", FileMode.Open, FileAccess.Read);
    byte[] barcodeFileStream = new byte[fileStream.Length];
    fileStream.Read(barcodeFileStream, 0, barcodeFileStream.Length);
    fileStream.Close();

    // Decodes barcode from the barcode file in byte array.
    TextResult[] aryResult = barcodeReader.DecodeFileInMemory(barcodeFileStream, string.Empty);
```

Useful Resources

Online Demo

- Read barcodes from images (on the server side)

API Documentation

- Windows Edition
- Linux Edition
- JavaScript Edition (Preview)
- iOS Edition
- Android Edition

Code Gallery

- Sample Code Download Page

Contact Us

- **Email:** support@dynamsoft.com
- **Online Chat:** http://www.dynamsoft.com/Support/LiveHelp.aspx
- **Telephone:** + 1 604-605-5491 | + 1 877-605-5491 (Toll-Free)
- **Fax:** 1-866-410-8856