www.bytescout.com

How to set image preprocessing filters with barcode reader sdk in C++ and ByteScout Barcode Suite

This code in C++ shows how to set image preprocessing filters with barcode reader sdk with this how to tutorial

The code displayed below will guide you to install an C++ app to set image preprocessing filters with barcode reader sdk. ByteScout Barcode Suite is the set that includes three different SDK products to generate barcodes, read barcodes and read and write spreadsheets: Barcode SDK, Barcode Reader SDK and Spreadsheet SDK. It can set image preprocessing filters with barcode reader sdk in C++.

These C++ code samples for C++ guide developers to speed up coding of the application when using ByteScout Barcode Suite. This C++ sample code is all you need for your app. Just copy and paste the code, add references (if needs to) and you are all set! Want to see how it works with your data then code testing will allow the function to be tested and work properly.

The trial version of ByteScout Barcode Suite can be downloaded for free from our website. It also includes source code samples for C++ and other programming languages.

Source Code Files:

ImagePreprocessingFilters.cpp

```
#include "stdafx.h"

///
/// This example demonstrates the use of image filters to improve the decoding confide
///

#import "c:\\Program Files\\Bytescout BarCode Reader SDK\\net2.00\\Bytescout.BarCodeRe

using namespace Bytescout_BarCodeReader;

void PrintFoundBarcodes(IReaderPtr pIReader);

int _tmain(int argc, _TCHAR* argv[])
{
        // Initialize COM.
        HRESULT hr = CoInitialize(NULL);

        // Create the interface pointer.
        IReaderPtr pIReader(__uuidof(Reader));

        // set the registration name and key
        _bstr_t registrationName(L"DEMO");
        pIReader->put_RegistrationName(registrationName);

        _bstr_t registrationKey(L"DEMO");
        pIReader->put_RegistrationKey(registrationKey);

        // Set barcode type to find
        _BarcodeTypeSelectorPtr pBarcodeTypesToFind;
        pIReader->get_BarcodeTypesToFind(&pBarcodeTypesToFind);
        pBarcodeTypesToFind->put_Code128(VARIANT_TRUE);


        // Get image filters collection
        _ImagePreprocessingFiltersCollection* pIImageFilters;
        pIReader->get_ImagePreprocessingFilters(&pIImageFilters);


        // WORKING WITH LOW CONTRAST BARCODE IMAGES

        pIImageFilters->AddContrast(40);

        // Add contrast adjustment for low-contrast image
        _bstr_t sampleFile1(L".\\low-contrast-barcode.png");
        wprintf(L"Image \"%s\"\n", static_cast<wchar_t*>(sampleFile1));

        pIReader->ReadFromFile(sampleFile1);
        PrintFoundBarcodes(pIReader);

        pIImageFilters->Clear();


        // WORKING WITH NOISY BARCODE IMAGES

        // Add median filter to lower a noise
```

```
        pIImageFilters->AddMedian();

        _bstr_t sampleFile2(L".\\noisy-barcode.png");
        wprintf(L"Image \"%s\"\n", static_cast<wchar_t*>(sampleFile2));

        pIReader->ReadFromFile(sampleFile2);
        PrintFoundBarcodes(pIReader);

        pIImageFilters->Clear();


        // WORKING WITH DENSE AND ILLEGIBLE BARCODES

        // Add the scale filter to enlarge the barcode to make gaps between bars more c
        pIImageFilters->AddScale_2(2.0);

        _bstr_t sampleFile3(L".\\too-dense-barcode.png");
        wprintf(L"Image \"%s\"\n", static_cast<wchar_t*>(sampleFile3));

        pIReader->ReadFromFile(sampleFile3);
        PrintFoundBarcodes(pIReader);

        pIImageFilters->Clear();


        // REMOVE EMPTY MARGINS FROM IMAGE TO SPEED UP THE PROCESSING

        // Add the crop filter to cut off empty margins from the image.
        // This will not improve the recognition quality but may speed up the processi
        // if you enabled multiple barcode types to search.
        pIImageFilters->AddCropDark();

        _bstr_t sampleFile4(L".\\barcode-with-large-margins.png");
        wprintf(L"Image \"%s\"\n", static_cast<wchar_t*>(sampleFile4));

        pIReader->ReadFromFile(sampleFile4);
        PrintFoundBarcodes(pIReader);

        pIImageFilters->Clear();



        // Uninitialize COM.
        CoUninitialize();

        // Wait until user press any key
        system("pause");

        return 0;
}

void PrintFoundBarcodes(IReaderPtr pIReader)
{
        // Get found barcode count
        long count;
        pIReader->get_FoundCount(&count);

        // Get found barcode information
        for (int i = 0; i < count; i++)
        {
```

```
                SymbologyType type;
                pIReader->GetFoundBarcodeType(i, &type);
                wprintf(L"Barcode type: %d\n", type);

                float confidence;
                pIReader->GetFoundBarcodeConfidence(i, &confidence);
                wprintf(L"Barcode confidence: %f\n", confidence);

                BSTR bstrValue;
                pIReader->GetFoundBarcodeValue(i, &bstrValue);
                wprintf(L"Barcode value: %s\n", bstrValue);
                ::SysFreeString(bstrValue);

                wprintf(L"\n");
        }

        if (count == 0)
        {
                wprintf(L"No barcodes found.\n");
        }
}
```

ImagePreprocessingFilters.sln

```
Microsoft Visual Studio Solution File, Format Version 12.00
# Visual Studio 2013
VisualStudioVersion = 12.0.40629.0
MinimumVisualStudioVersion = 10.0.40219.1
Project("{8BC9CEB8-8B4A-11D0-8D11-00A0C91BC942}") = "ImagePreprocessingFilters", "Image
EndProject
Global
        GlobalSection(SolutionConfigurationPlatforms) = preSolution
                Debug|Win32 = Debug|Win32
                Release|Win32 = Release|Win32
        EndGlobalSection
        GlobalSection(ProjectConfigurationPlatforms) = postSolution
                {74A23FC0-E323-4980-9363-41326A457785}.Debug|Win32.ActiveCfg = Debug|W
                {74A23FC0-E323-4980-9363-41326A457785}.Debug|Win32.Build.0 = Debug|Win
                {74A23FC0-E323-4980-9363-41326A457785}.Release|Win32.ActiveCfg = Releas
                {74A23FC0-E323-4980-9363-41326A457785}.Release|Win32.Build.0 = Release
        EndGlobalSection
        GlobalSection(SolutionProperties) = preSolution
                HideSolutionNode = FALSE
        EndGlobalSection
EndGlobal
```

stdafx.cpp

```
// stdafx.cpp : source file that includes just the standard includes
// CPPExample.pch will be the pre-compiled header
// stdafx.obj will contain the pre-compiled type information

#include "stdafx.h"

// TODO: reference any additional headers you need in STDAFX.H
// and not in this file
```

stdafx.h

```
// stdafx.h : include file for standard system include files,
// or project specific include files that are used frequently, but
// are changed infrequently
//

#pragma once

#include "targetver.h"

#include <stdio.h>
#include <tchar.h>



// TODO: reference additional headers your program requires here
```

targetver.h

```
#pragma once
```

```
// Including SDKDDKVer.h defines the highest available Windows platform.

// If you wish to build your application for a previous Windows platform, include WinSD
// set the _WIN32_WINNT macro to the platform you wish to support before including SDKD

#include <SDKDDKVer.h>
```

VIDEO

https://www.youtube.com/watch?v=NEwNs2b9YN8

ON-PREMISE OFFLINE SDK

60 Day Free Trial or Visit ByteScout Barcode Suite Home Page
Explore ByteScout Barcode Suite Documentation
Explore Samples
Sign Up for ByteScout Barcode Suite Online Training

ON-DEMAND REST WEB API

Get Your API Key
Explore Web API Docs
Explore Web API Samples

visit www.ByteScout.com

visit www.PDF.co