

www.bytescout.com

wpf barcode control for desktop with barcode sdk in C# and ByteScout Barcode Suite

Learn wpf barcode control for desktop with barcode sdk in C#

Easy to understand coding instructions are written to assist you to try-out the features without the requirement to write your own code. ByteScout Barcode Suite was made to help with wpf barcode control for desktop with barcode sdk in C#. ByteScout Barcode Suite is the bundle that provides 3 SDK products to generate barcodes (Barcode SDK), read barcodes (Barcode Reader SDK) and read and write spreadsheets (Spreadsheet SDK).

C# code snippet like this for ByteScout Barcode Suite works best when you need to quickly implement wpf barcode control for desktop with barcode sdk in your C# application. Follow the steps-by-step instructions from the scratch to work and copy and paste code for C# into your editor. These C# sample examples can be used in one or many applications.

On our website you may get trial version of ByteScout Barcode Suite for free. Source code samples are included to help you with your C# application.

FOR MORE INFORMATION AND FREE TRIAL:

[Download Free Trial SDK \(on-premise version\)](#)

[Read more about ByteScout Barcode Suite](#)

[Explore API Documentation](#)

[Get Free Training for ByteScout Barcode Suite](#)

[Get Free API key for Web API](#)

[visit \[www.ByteScout.com\]\(http://www.ByteScout.com\)](#)

Source Code Files:

App.xaml.cs

```
using System;
using System.Collections.Generic;
using System.Configuration;
using System.Data;
using System.Linq;
using System.Windows;

namespace Bytescout.BarCode.WPFDemo
{
    /// <summary>
    /// Interaction logic for App.xaml
    /// </summary>
    public partial class App : Application
    {
    }
}
```

MainWindow.xaml.cs

```
using System;
using System.Windows;
using System.Windows.Controls;
using System.Windows.Media.Imaging;

namespace Bytescout.BarCode.WPFDemo
{
    /// <summary>
    /// Interaction logic for MainWindow.xaml
    /// </summary>
    public partial class MainWindow : Window
    {

        #region Constants

        private const int BarHeight = 50;
        private const int PdfBarHeight = 6;

        #endregion

        #region Constructor
        /// <summary>
        /// Initializes a new instance of the <see cref="MainWindow"/> class.
        /// </summary>
        public MainWindow()
        {
            InitializeComponent();
        }
    }
}
```

```
}

#endregion

#region Controls event handlers
/// <summary>
/// Handles the SelectionChanged event of the cmbSymbologyType control.
/// </summary>
/// <param name="sender">The source of the event.</param>
/// <param name="e">The <see cref="System.Windows.Controls.SelectionChangedEventArgs"/> instance of the event arguments to handle.
private void cmbSymbologyType_SelectionChanged(object sender, SelectionChangedEventArgs e)
{
    UpdateBarcode();
}

/// <summary>
/// Handles the Click event of the btnGenerate control.
/// </summary>
/// <param name="sender">The source of the event.</param>
/// <param name="e">The <see cref="System.Windows.RoutedEventArgs"/> instance of the event arguments to handle.
private void btnGenerate_Click(object sender, RoutedEventArgs e)
{
    UpdateBarcode();
}

/// <summary>
/// Handles the Click event of the btnSaveToFile control.
/// </summary>
/// <param name="sender">The source of the event.</param>
/// <param name="e">The <see cref="System.Windows.RoutedEventArgs"/> instance of the event arguments to handle.
private void btnSaveToFile_Click(object sender, RoutedEventArgs e)
{
    SaveToFile();
}

#endregion

#region Menu items event handlers
/// <summary>
/// Handles the Click event of the mnuSaveToFile control.
/// </summary>
/// <param name="sender">The source of the event.</param>
/// <param name="e">The <see cref="System.Windows.RoutedEventArgs"/> instance of the event arguments to handle.
private void mnuSaveToFile_Click(object sender, RoutedEventArgs e)
{
    SaveToFile();
}

/// <summary>
/// Handles the Click event of the mnuExit control.
/// </summary>
/// <param name="sender">The source of the event.</param>
/// <param name="e">The <see cref="System.Windows.RoutedEventArgs"/> instance of the event arguments to handle.
private void mnuExit_Click(object sender, RoutedEventArgs e)
{
    Close();
}

/// <summary>
/// Handles the Click event of the mnuCopy control.
/// </summary>
```

```

    ///> The source of the event.</param>
    ///> The <see cref="System.Windows.RoutedEventArgs"/> instance
private void mnuCopy_Click(object sender, RoutedEventArgs e)
{
    BitmapSource barcode = ctrlBarcodeControl.GetImage();
    Clipboard.SetImage(barcode);
    barcode = null;
}

///> Handles the Click event of the mnuBarcodeHome control.
///>
private void mnuBarcodeHome_Click(object sender, RoutedEventArgs e)
{
    System.Diagnostics.Process.Start("http://bytescout.com/bytescoutbarcodesdk");
}

///> Handles the Click event of the mnuHelp control.
///>
private void mnuHelp_Click(object sender, RoutedEventArgs e)
{
}

///> Handles the TextChanged event of the txtValueToEncode control.
///>
private void txtValueToEncode_TextChanged(object sender, TextChangedEventArgs e)
{
    UpdateBarcode();
}

///> Handles the TextChanged event of the txtSupplementalValue control.
///>
private void txtSupplementalValue_TextChanged(object sender, TextChangedEventArgs e)
{
    UpdateBarcode();
}
#endregion

#region Private implementation

public object[] GetObjectsFromEnum()
{
    object[] objArray = new object[Enum.GetValues(typeof(SymbologyType)).Length];
    for (int i = 0; i < objArray.Length; i++)
    {
        objArray[i] = ((SymbologyType)Enum.GetValues(typeof(SymbologyType)).Get
    }
    return objArray;
}

```

```

}

private void UpdateBarcode()
{
    SymbologyType symbology = (SymbologyType)Enum.GetValues(typeof(SymbologyType));
    txtSymbologyDescription.Text = ctrlBarcodeControl.GetValueRestrictions(symbology);

    try
    {
        if (symbology == SymbologyType.EAN13 || symbology == SymbologyType.ISBN)
        {
            txtSupplementalValue.IsEnabled = true;
            lblSupplementalValue.IsEnabled = true;
            txtSymbologyDescription.Text += " " + ctrlBarcodeControl.GetSupplementalValue();
        }
        else
        {
            txtSupplementalValue.IsEnabled = false;
            lblSupplementalValue.IsEnabled = false;
        }

        lblErrorMessage.Content = "";
        ctrlBarcodeControl.RegistrationKey = "XXXXXXXXXXXXXXXXXXXX";
        ctrlBarcodeControl.RegistrationName = "YYYYYYYYYYYYYYYYYYYY";
        ctrlBarcodeControl.Symbology = symbology;
        ctrlBarcodeControl.SupplementValue = txtSupplementalValue.Text;
        ctrlBarcodeControl.Value = txtValueToEncode.Text;
        ctrlBarcodeControl.DrawCaptionFor2DBarcodes = chkDrawCaptionFor2D.IsChecked;
        ctrlBarcodeControl.AutoFitToControlSize = chkAutoFitToContainer.IsChecked;
        ctrlBarcodeControl.Caption = "";

        if (symbology == SymbologyType.PDF417 || symbology == SymbologyType.PDF417_2D ||
            symbology == SymbologyType.MacroPDF417 || symbology == SymbologyType.QRCode ||
            symbology == SymbologyType.GS1_DataMatrix)
        {
            ctrlBarcodeControl.BarHeight = PdfBarHeight;
        }
        else if (symbology == SymbologyType.MicroPDF417)
        {
            ctrlBarcodeControl.BarHeight = PdfBarHeight / 2;
        }
        else
        {
            ctrlBarcodeControl.BarHeight = BarHeight;
        }
    }
    catch (Exception)
    {
        lblErrorMessage.Content = "Value is invalid for current symbology. Please try again.";
    }
}

private void SaveToFile()
{
    Microsoft.Win32.SaveFileDialog dlg = new Microsoft.Win32.SaveFileDialog();
    dlg.Filter = "PNG Image (*.png)|TIFF Image (*.tif)|*.tiff|JPEG image (*.jpg)|*.jpeg";
    dlg.ValidateNames = true;
    dlg.FilterIndex = 1;
    dlg.OverwritePrompt = true;
    dlg.CheckPathExists = true;
}

```

```

dlg.AddExtension = true;

bool? result = dlg.ShowDialog(this);
if (result.HasValue && result.Value)
{
    try
    {
        if (System.IO.Path.GetExtension(dlg.FileName).ToLowerInvariant() == ".emf")
            throw new BarcodeException("Saving as EMF is disabled.\nYou should save as PDF or XPS instead.");

        if (chkCutUnusedSpace.IsChecked.Value)
        {
            bool cut = ctrlBarcodeControl.CutUnusedSpace;
            ctrlBarcodeControl.CutUnusedSpace = true;
            ctrlBarcodeControl.SaveImage(dlg.FileName);
            ctrlBarcodeControl.CutUnusedSpace = cut;
        }
        else
        {
            ctrlBarcodeControl.SaveImage(dlg.FileName);
        }
    }
    catch (System.Exception e)
    {
        MessageBox.Show(e.Message);
    }
}
}

#endregion

#region Main window event handlers
private void Window_Loaded(object sender, RoutedEventArgs e)
{
    foreach (object o in GetObjectsFromEnum())
    {
        this.cboSymbologyType.Items.Add(o);
    }
    this.cboSymbologyType.SelectedIndex = 0;
}
#endregion

}
}

```

<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](#)

[www.bytescout.com](#)