

[www.bytescout.com](http://www.bytescout.com)

## WPF barcode control for web in C# with ByteScout Barcode SDK

### Tutorial: how to do WPF barcode control for web in C#

ByteScout tutorials explain the material for programmers who use C#. ByteScout Barcode SDK helps with WPF barcode control for web in C#. ByteScout Barcode SDK is the robust library (Software Development Kit) that is designed for automatic generation of high-quality barcodes for printing, electronic documents and pdf. All popular barcode types are supported from Code 39 and Code 129 to QR Code, UPC, GS1, GS-128, Datamatrix, PDF417, Maxicode and many others. Provides support for full customization of fonts, colors, output and printing sizes. Special tools are included to verify output quality and printing quality. Can add generated barcode into new or existing documents, images and PDF.

The SDK samples like this one below explain how to quickly make your application do WPF barcode control for web in C# with the help of ByteScout Barcode SDK. To do WPF barcode control for web in your C# project or application you may simply copy & paste the code and then run your app! This basic programming language sample code for C# will do the whole work for you in implementing WPF barcode control for web in your app.

Free trial version of ByteScout Barcode SDK is available on our website. Get it to try other samples for C#.

FOR MORE INFORMATION AND FREE TRIAL:

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

[Read more about ByteScout Barcode SDK](#)

[Explore API Documentation](#)

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

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

[visit 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;
using System.Windows.Navigation;

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

## MainPage.xaml.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Windows;
using System.Windows.Controls;
using System.Windows.Data;
using System.Windows.Documents;
using System.Windows.Input;
using System.Windows.Media;
using System.Windows.Media.Imaging;
using System.Windows.Navigation;
using System.Windows.Shapes;
using Bytescout.BarCode.WPF;
using Bytescout.BarCode;

namespace Bytescout.BarCode.WebDemo
{
    /// <summary>
    /// Interaction logic for MainPage.xaml
    /// </summary>
    public partial class MainPage : Page
    {
        #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 MainPage()
{
    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.
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.
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.
private void btnSaveToFile_Click(object sender, RoutedEventArgs e)
{
    SaveToFile();
}

/// <summary>
/// Handles the Checked event of the chkDrawCaptionFor2D 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.
private void chkDrawCaptionFor2D_Checked(object sender, RoutedEventArgs e)
{
    ctrlBarcodeControl.DrawCaptionFor2DBarcodes = chkDrawCaptionFor2D.IsChecked;
}

/// <summary>
/// Handles the Checked event of the chkAutoFitToContainer 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.
private void chkAutoFitToContainer_Checked(object sender, RoutedEventArgs e)
```

```

private void chkAutoFitToContainer_Checked(object sender, RoutedEventArgs e)
{
    ctrlBarcodeControl.AutoFitToControlSize = chkAutoFitToContainer.IsChecked;
}

/// <summary>
/// Handles the Checked event of the chkCutUnusedSpace control.
/// </summary>
/// <param name="sender">The source of the event.</param>
/// <param name="e">The <see cref="System.Windows.RoutedEventArgs"/> instance containing event data.
private void chkCutUnusedSpace_Checked(object sender, RoutedEventArgs e)
{
}

/// <summary>
/// Handles the TextChanged event of the txtValueToEncode control.
/// </summary>
/// <param name="sender">The source of the event.</param>
/// <param name="e">The <see cref="System.Windows.Controls.TextChangedEventArgs"/> instance containing event data.
private void txtValueToEncode_TextChanged(object sender, TextChangedEventArgs e)
{
    UpdateBarcode();
}

/// <summary>
/// Handles the TextChanged event of the txtSupplementalValue control.
/// </summary>
/// <param name="sender">The source of the event.</param>
/// <param name="e">The <see cref="System.Windows.Controls.TextChangedEventArgs"/> instance containing event data.
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)).GetItem(i));
    }
    return objArray;
}

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

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

```

```

        }

        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.PDI)
            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)|JPEG image (*.jpg)|All files (*.*)";
    dlg.ValidateNames = true;
    dlg.FilterIndex = 1;
    dlg.OverwritePrompt = true;
    dlg.CheckPathExists = true;
    dlg.AddExtension = true;

    bool? result = dlg.ShowDialog(Application.Current.MainWindow);
    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 a standard image file type such as PNG or JPEG.");

            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 Page_Loaded(object sender, RoutedEventArgs e)
{
    foreach (object o in GetObjectsFromEnum())
    {
        this.cboSymbologyType.Items.Add(o);
    }
    this.cboSymbologyType.SelectedIndex = 0;
}
#endregion
}
}

```

---

## VIDEO

<https://www.youtube.com/watch?v=REnj3A-oSPI>

## ON-PREMISE OFFLINE SDK

[60 Day Free Trial](#) or [Visit ByteScout Barcode SDK Home Page](#)  
[Explore ByteScout Barcode SDK Documentation](#)  
[Explore Samples](#)  
[Sign Up for ByteScout Barcode SDK 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](#)