Chapter 41: WinForms SpellCheckBox

Example on how to add a spell check box to a WindowsForms application. This example DOES NOT require Word to be installed nor does it use Word in any way.

It uses WPF Interop using the ElementHost control to create a WPF UserControl from a WPF TextBox. WPF TextBox has a built in function for spell check. We are going to leverage this built in function rather than relying on an external program.

Section 41.1: ElementHost WPF TextBox

This example is was modeled after an example that I found on the internet. I can't find the link or I would give the author credit. I took the sample that I found and modified it to work for my application.

1. Add the following references:

> System.Xaml, PresentationCore, PresentationFramework, WindowsBase, and WindowsFormsIntegration

2. Create a new Class and past this code

```
Imports System
Imports System.ComponentModel
Imports System.ComponentModel.Design.Serialization
Imports System.Windows
Imports System.Windows.Controls
Imports System.Windows.Forms.Integration
Imports System.Windows.Forms.Design

<Designer(GetType(ControlDesigner))> _
Class SpellCheckBox
Inherits ElementHost

Private box As TextBox

Public Sub New()
    box = New TextBox()
    MyBase.Child = box
    AddHandler box.TextChanged, AddressOf box_TextChanged
    box.SpellCheck.IsEnabled = True
    box.VerticalScrollBarVisibility = ScrollBarVisibility.Auto
    Me.Size = New System.Drawing.Size(100, 20)
End Sub

Private Sub box_TextChanged(ByVal sender As Object, ByVal e As EventArgs)
    OnTextChanged(EventArgs.Empty)
End Sub

<DefaultValue("")> _
Public Overrides Property Text() As String
    Get
        Return box.Text
    End Get
    Set(ByVal value As String)
        box.Text = value
    End Set
End Property
```

```
    <DefaultValue(True)> _
    Public Property MultiLine() As Boolean
        Get
            Return box.AcceptsReturn
        End Get
        Set(ByVal value As Boolean)
            box.AcceptsReturn = value
        End Set
    End Property

    <DefaultValue(True)> _
    Public Property WordWrap() As Boolean
        Get
            Return box.TextWrapping <> TextWrapping.Wrap
        End Get
        Set(ByVal value As Boolean)
            If value Then
                box.TextWrapping = TextWrapping.Wrap
            Else
                box.TextWrapping = TextWrapping.NoWrap
            End If
        End Set
    End Property

    <DesignerSerializationVisibility(DesignerSerializationVisibility.Hidden)> _
    Public Shadows Property Child() As System.Windows.UIElement
        Get
            Return MyBase.Child
        End Get
        Set(ByVal value As System.Windows.UIElement)
            '' Do nothing to solve a problem with the serializer !!
        End Set
    End Property

    End Class
```

3. Rebuild the solution.
4. Add a new form.
5. Search the toolbox for your Class name. This example is "SpellCheck". It should be listed under 'YourSoulutionName' Components.
6. Drag the new control to your form
7. Set any of the mapped properties in the forms load event

```
Private Sub form1_Load(sender As Object, e As EventArgs) Handles Me.Load
    spellcheckbox.WordWrap = True
    spellcheckbox.MultiLin = True
    'Add any other property modifiers here...
End Sub
```

7. The last thing that you need to do is to change the DPI Awareness of your application. This is because you are using WinForms application. By default all WinForms applications are DPI UNAWARE. Once you execute a control that has an element host (WPF Interop), the application will now become DPI AWARE. This may or may not mess with your UI Elements. The solution to this is to FORCE the application to become DPI UNAWARE. There are 2 ways to do this. The first is through the manifest file and the second is to hard code it in to your program. If you are using OneClick to deploy your application, you must hard code it, not use the manifest file or errors will be inevitable.

Both of the following examples can be found at the following: [WinForms Scaling at Large DPI Settings - Is It Even Possible?] Thanks to Telerik.com for the great explanation on DPI.

> Hard coded DPI Aware code example. This MUST be executed before the first form is initialized. I always place this in the ApplicationEvents.vb file. You can get to this file by right clicking on your project name in the solution explorer and choosing "Open". Then choose the application tab on the left and then click on "View Application Events" on the lower right next to the splash screen drop down.

```
Namespace My

    ' The following events are available for MyApplication:
    '
    ' Startup: Raised when the application starts, before the startup form is created.
    ' Shutdown: Raised after all application forms are closed.  This event is not raised if the
application terminates abnormally.
    ' UnhandledException: Raised if the application encounters an unhandled exception.
    ' StartupNextInstance: Raised when launching a single-instance application and the application is
already active.
    ' NetworkAvailabilityChanged: Raised when the network connection is connected or disconnected.
    Partial Friend Class MyApplication

    Private Enum PROCESS_DPI_AWARENESS
        Process_DPI_Unaware = 0
        Process_System_DPI_Aware = 1
        Process_Per_Monitor_DPI_Aware = 2
    End Enum

    Private Declare Function SetProcessDpiAwareness Lib "shcore.dll" (ByVal Value As
PROCESS_DPI_AWARENESS) As Long

    Private Sub SetDPI()
        'Results from SetProcessDPIAwareness
        'Const S_OK = &H0&
        'Const E_INVALIDARG = &H80070057
        'Const E_ACCESSDENIED = &H80070005

        Dim lngResult As Long

        lngResult = SetProcessDpiAwareness(PROCESS_DPI_AWARENESS.Process_DPI_Unaware)

    End Sub

    Private Sub MyApplication_Startup(sender As Object, e As ApplicationServices.StartupEventArgs)
Handles Me.Startup
        SetDPI()
    End Sub

End Namespace
```

> Manifest Example

```
<assembly xmlns="urn:schemas-microsoft-com:asm.v1" manifestVersion="1.0" xmlns:asmv3="urn:schemas-
microsoft-com:asm.v3" >
   <asmv3:application>
        <asmv3:windowsSettings xmlns="http://schemas.microsoft.com/SMI/2005/WindowsSettings">
             <dpiAware>true</dpiAware>
```

```
        </asmv3:windowsSettings>
    </asmv3:application>
</assembly>
```