XAML Do-It-Yourself シリーズ

第 9 回 ユーザー コントロール

Microsoft®

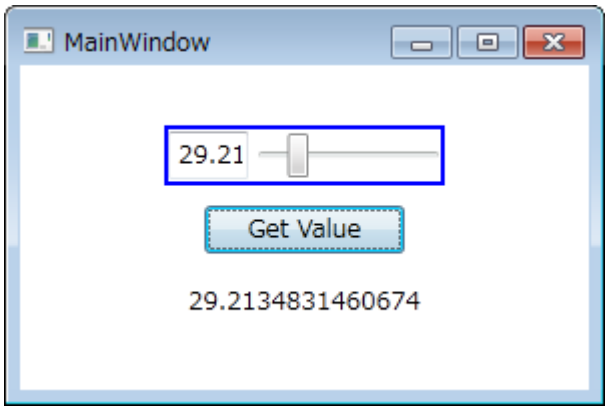
XAML Do-It-Yourself
第 9 回 ユーザー コントロール

XAML Do-It-Yourself 第 9 回は、ユーザー コントロールについて学習します。ユーザー コントロールとは、WPF で提供されているコントロールを組み合わせて作成した独自のコントロールです。

今回は次の 2 点について学習します。

・ユーザー コントロールの作成方法
・WPF アプリケーションでユーザー コントロールを利用する方法

ユーザー コントロールを利用したアプリケーションの一例を次に示します。青い枠線で囲まれた部分がユーザー コントロールです。



これは TextBox と Slider (スライダー コントロール) を組み合わせて作成しており、Slider から値を設定することも、TextBox から直接数値を入力することもできます。アプリケーションでは、ボタンがクリックされたら、ボタンの下のラベルに、ユーザー コントロールに設定された値を表示します。

■ユーザー コントロールの作成

ユーザー コントロールは、UserControl クラス あるいは Control クラスを継承して作成できます。どちらのクラスを継承してコントロールを作成するかにより、XAML での記述方法も変わってきます。

UserControl クラスを継承したユーザー コントロールは、第 7 回で紹介したテンプレートをサポートしませんが、作成が容易であるという特徴があります。今回は、UserControl クラスを継承し、複数のコントロールを組み合わせた独自のコントロールを作成します。

テンプレートをサポートしたい場合や、作成したユーザー コントロールのカスタマイズが必要な場合には、UserControl クラスでなく、Control クラスを継承する必要があります。

Visual Studio 2010 を用いてユーザー コントロールを作成するには、プロジェクト テンプレートとして「WPF ユーザー コントロール ライブラリ」を選択してプロジェクトを作成します。この場合に自動生成されるコードは、UserControl を継承したユーザー コントロールのひな型になります (プロジェクト名は MyWpfControlLibrary とします)。

●ルート要素は UserControl

これまで扱ってきた XAML のコードでは、ルート要素として <Window> 要素を用いていました。これに対して UserControl を継承するユーザー コントロールでは、<UserControl> 要素がルート要素となります。

以降では冒頭で示したような、TextBox と Slider を組み合わせた、キーボードでもマウスでも数値の入力が行えるユーザー コントロールを作成していきます。

●XAML の記述

上述の手順で、ユーザー コントロールのルート要素には <UserControl> を持つ UserControl1.xaml が生成されます。

```
<UserControl x:Class="MyWpfControlLibrary.UserControl1"
        xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
        xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
        xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
        xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
        Height="30" Width="140">

</UserControl>
```

<UserControl> 要素の記述は、通常の WPF アプリケーションにおける <Window> 要素と変わりはありません。上記のように Height (高さ) と Width (幅) を使ってコントロールの基本的な大きさを指定します。x:Class 属性には、コード ビハインド クラスが指定されています。

以下のリストのように、その内容の記述も、通常の WPF アプリケーションでコントロールを配置する際の記述と特に変わりはありません。ここではユーザー コントロールの境界が分かりやすいように <Border> 要素で青色の外枠を作成し、その中に <StackPanel> 要素を配置して、この中に <TextBox> 要素と <Slider> 要素を配置しています。

このとき、TextBox の Text プロパティを Slider の Value プロパティにバインドしておきます。これにより Slider の操作と同期して TextBox の内容が変化します。またバインドの設定に "TwoWay" を指定することにより、TextBox に入力された値が Slider に反映されるようにします。

```
<UserControl x:Class="MyWpfControlLibrary.UserControl1"
   xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
   xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
   Height="30" Width="140">
  <Border BorderBrush="Blue" BorderThickness="2" >
```

```
    <StackPanel Orientation="Horizontal" >
      <TextBox HorizontalAlignment="Left" Width="40" VerticalAlignment="Center">
        <TextBox.Text>
          <Binding ElementName="mySlider" Path="Value" Mode="TwoWay"/>
        </TextBox.Text>
      </TextBox>
      <Slider  Name="mySlider" Width="100" VerticalAlignment="Center"
Maximum="100" SmallChange="1" />
    </StackPanel>
  </Border>
</UserControl>
```

<Slider> 要素には "mySlider" という名前を付けています。この名前はコード ビハインドのロジックから Slider の値を参照するために利用します。

XAML の記述は以上です。次にコード ビハインド ロジックを記述して、ユーザー コントロールのプロパティなどを作成します。

●コード ビハインド ロジックの作成

<UserControl> 要素の x:Class 属性で、コード ビハインド クラスが "MyWpfControlLibrary.UserControl1" と記述されているとおり、ソースコード（UserControl1.xaml.cs）には MyWpfControlLibrary 名前空間に UserControl1 というクラスが作成されています。

ここで、UserControl1 クラスは UserControl の派生クラスとして宣言されています。この UserControl1 クラスでは、コンストラクターといくつかのプロパティを用意します。

まずコンストラクターでは、コンポーネントの初期化と、Slider の最大値と最小値を設定しておきます。

プロパティとして公開するのは、コントロールの値の最小値を設定 (取得) する Minimum、最大値を設定 (取得) する Maximum、現在選択されている値を設定 (取得) する Value の 3 つです。

```
using System.Windows.Controls;

namespace MyWpfControlLibrary
{
  public partial class UserControl1 : UserControl
  {
    public UserControl1()
    {
      InitializeComponent()
    }

    public double Maximum {
      set { mySlider.Maximum = value; }
      get { return mySlider.Maximum; }
    }

    public double Minimum
```

```
    {
      set { mySlider.Minimum = value; }
      get { return mySlider.Minimum; }
    }

    public double Value
    {
      set { mySlider.Value = value; }
      get { return mySlider.Value; }
    }
  }
}
```

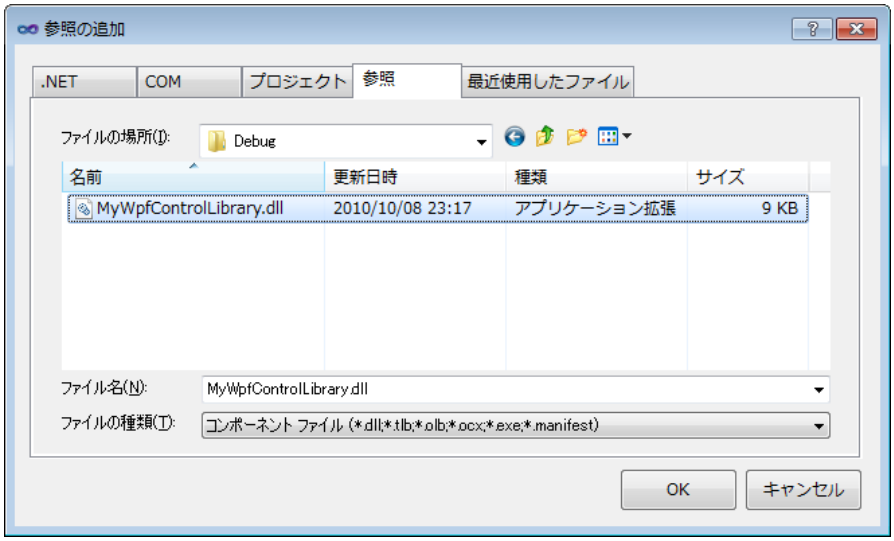
以上の作業で UserControl クラスを継承したユーザー コントロールが完成です。プロジェクトをビルドすると、ライブラリ (DLL ファイル) が生成されます。作成したユーザー コントロールを利用するアプリケーションは、このライブラリを利用します。ここではプロジェクト名を ”MyWpfControlLibrary” としたため、最終的に ”MyWpfControlLibrary.dll” という DLL ファイルが得られます。

■ユーザー コントロールの使用

続いては、いま作成したユーザー コントロールを利用するアプリケーションを作成します。

Visual Studio 2010 を使って、ユーザー コントロールを利用した WPF アプリケーションを作成する場合は、メニューから[プロジェクト]–「参照の追加」を選択してプロジェクトにユーザー コントロールを追加し、さらにツールボックスにコントロールを追加します。これによりツールボックスからドラッグ&ドロップでユーザー コントロールをウィンドウに配置できるようになります。

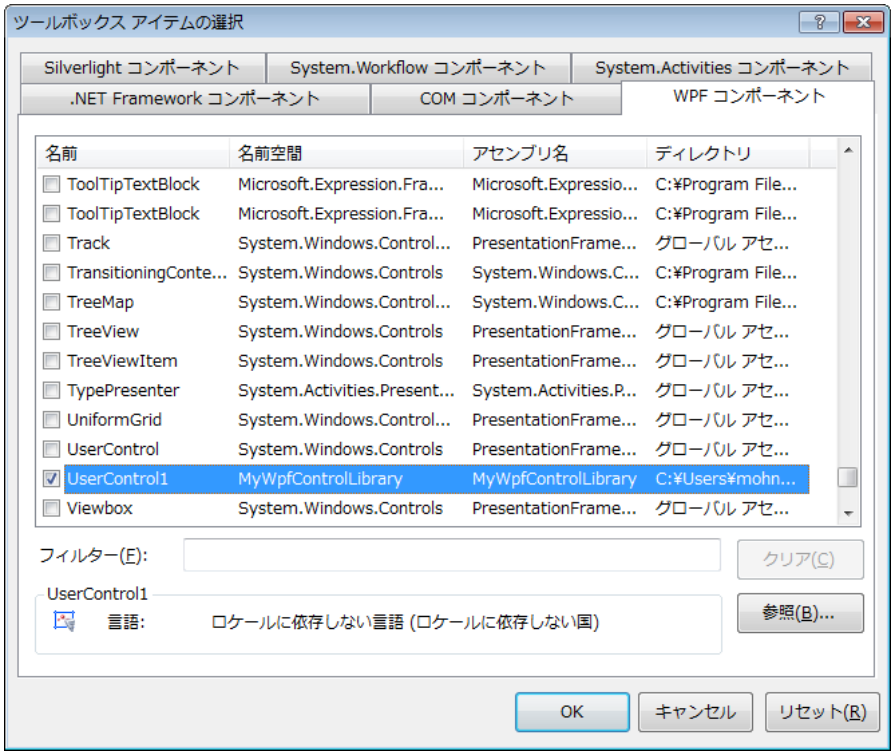
参照の追加



Visual Studio 2010 のツールボックスにユーザー コントロールを追加するには、ツールボックス上で右クリックしてコンテキスト メニューを開き [アイテムの選択] を選択します。

[ツールボックス アイテムの選択]ダイアログ ボックスが開くので、[参照]ボタンをクリックして、作成したユーザー コントロールを選択します。これでツールボックスにユーザー コントロールが追加されます。

ツール ボックス にユーザー コントロールを追加



ユーザー コントロールを利用する WPF アプリケーションでは、XAML ファイル内でユーザー コントロールの使用を明示します。まず <Window> 要素で、ユーザー コントロールの名前空間を追加します (ツールボックスからコントロールをドラッグ アンド ドロップする場合は、自動的に追加されます)。

```
<Window x:Class="WpfApplication.Window1"
    xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
    xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
    xmlns:my="clr-namespace:MyWpfControlLibrary;assembly=MyWpfControlLibrary"
    Title="MainWindow" Height="200" Width="300">
```

ここでは xmlns:my 属性を記述して、MyWpfControlLibrary というアセンブリの MyWpfControlLibrary 名前空間を "my" という名前で参照できるようにしています。これにより、ユーザー コントロールは、<my:UserControl1> という具合に記述できます。

```
<Window x:Class="WpfApplication9.MainWindow"
    xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
    xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
    xmlns:my="clr-namespace:MyWpfControlLibrary;assembly=MyWpfControlLibrary"
```

```
    Title="MainWindow" Height="200" Width="300">
  <StackPanel VerticalAlignment="Center">
   <my:UserControl1 Name="userControl" VerticalAlignment="Bottom" Minimum="10" />
   <Button Width="100" Margin="0 10 0 10" Click="Button_Click">Get Value</Button>
   <Label Name="label1" Content="-" HorizontalAlignment="Center"/>
  </StackPanel>
</Window>
```

<my:UserControl1> 要素では、ユーザー コントロールの最小値を表す Minimum 属性に 10 を設定していますが、この Minimum 属性は、先ほどコード ビハインド ロジックで作成した Minimum プロパティです。

ここでは、ユーザー コントロールに加えて、Button と Label を配置し、<Button> 要素に Click 属性を記述して、Button がクリックされたら、コード ビハインド ロジックの Button_Click メソッドが実行されるようにしています。

Button_Click メソッドでは、ユーザー コントロールの Value プロパティを参照して現在設定されている値を取得し、この内容を Label にセットします。

```
using System.Windows;

namespace WpfApplication
{
  public partial class Window1 : Window
  {
    public Window1()
    {
      InitializeComponent();
    }

    private void Button_Click( object sender, RoutedEventArgs e )
    {
      double v = userControl.Value;
      label1.Content = v.ToString();
    }
  }
}
```

■まとめ

今回は、ユーザー コントロールの作成方法を紹介しました。既存のコントロールをうまく組み合わせることにより、より便利な独自のコントロールを作り出せます。また、そのようにして作成したコントロールは通常のコントロールと同様に利用でき、大切な資産となります。

次回は、WPF アプリケーションでメディアを扱う方法を紹介します。