


Drag And Drop Item in ListBox in WPF

Posted by [Diptimaya Patra](#) in [Articles](#) | [WPF](#) on January 17, 2010

Tags: [Drag And Drop Item](#), [ListBox In WPF](#), [WPF](#)

In this article we will see how we achieve Drag and Drop behaviour for ListBox Item.

Source: <http://www.c-sharpcorner.com/uploadfile/dpatra/drag-and-drop-item-in-listbox-in-wpf/>

Reader Level: 



Download Files:

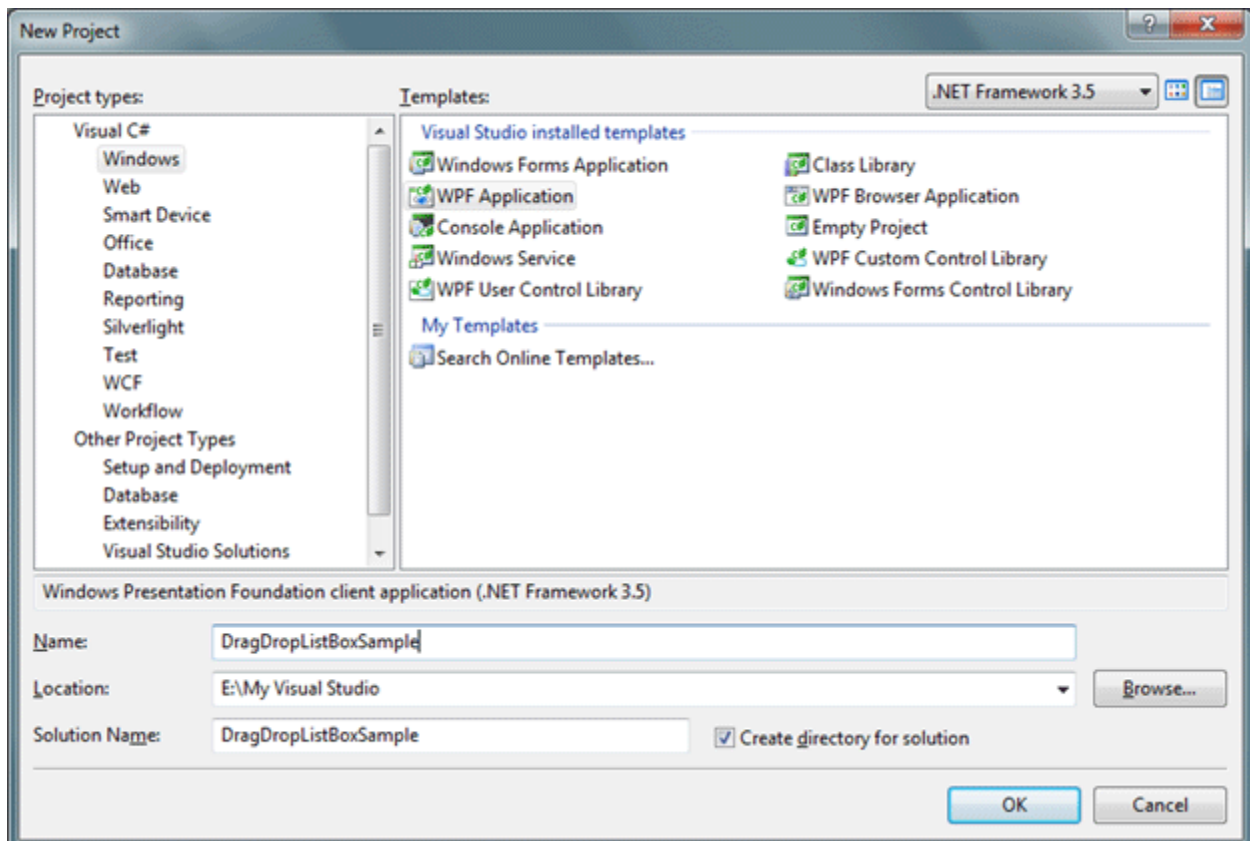
- [DragDropListBoxSample.zip](#)

Introduction

In this article we will see how we achieve Drag and Drop behaviour for ListBox Item.

Creating WPF Project

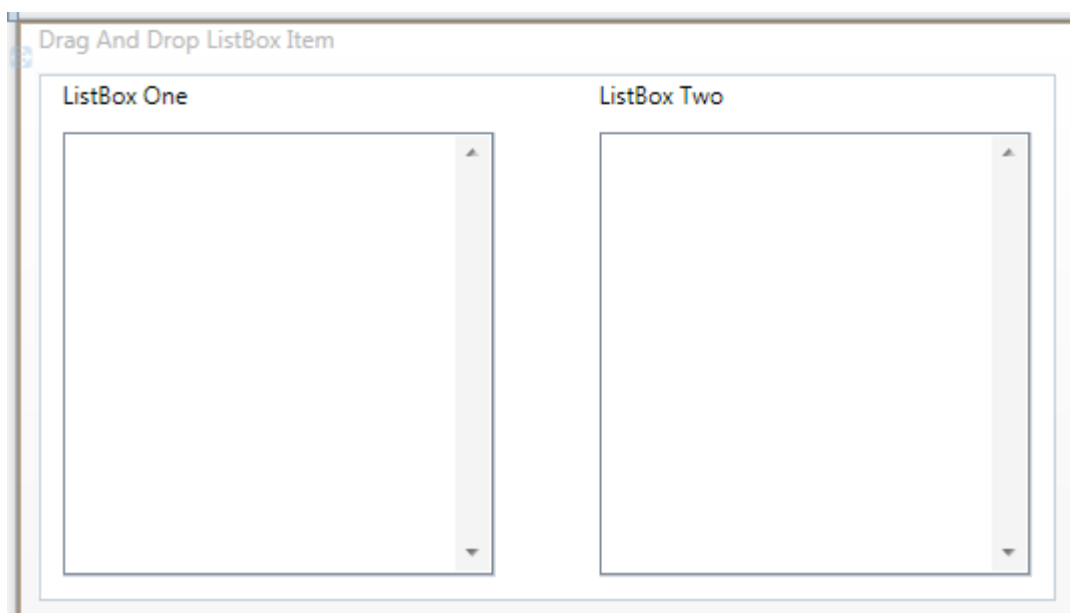
Fire up Visual Studio 2008 and create a new WPF Project. Name it as DragDropListBoxSample.



Basic idea of our sample application is to have two ListBox and we would provide Drag item from First ListBox and Drop into the Second ListBox.

So let's have two ListBox and name the ListBoxes as lbOne, lbTwo.

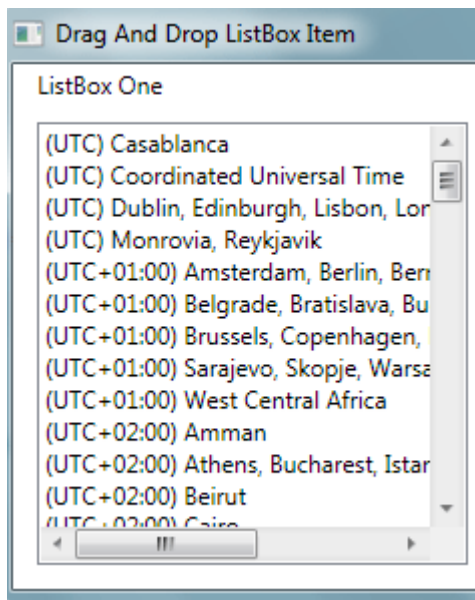
The following is the basic design how our application would look like.



Here what we would list in the First ListBox. A list of TimeZones from the class TimeZoneInfo.

```
ObservableCollection<string> zoneList = new ObservableCollection<string>();
```

```
public Window1()
{
    InitializeComponent();
    foreach (TimeZoneInfo tzi in TimeZoneInfo.GetSystemTimeZones())
    {
        zoneList.Add(tzi.ToString());
    }
    lbOne.ItemsSource = zoneList;
}
```



As you see our first listbox is loaded with all TimeZones.

Now to achieve Drag and Drop we need to make AllowDrop="True" for our target ListBox; and we need PreviewMouseLeftButtonDown event for the Source and Drop event for Target ListBoxes respectively.

Follow the below XAML code to see the events and properties.

```
<Window x:Class="DragDropListBoxSample.Window1"
    xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
    xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
    Title="Drag And Drop ListBox Item" Height="300" Width="529">
    <Grid>
        <ListBox x:Name="lbOne"
            PreviewMouseLeftButtonDown="ListBox_PreviewMouseLeftButtonDown"
            HorizontalAlignment="Left" Margin="12,29,0,12" Width="215"
            ScrollViewer.VerticalScrollBarVisibility="Visible" />
```

```

<ListBox x:Name="lbTwo" Drop="ListBox_Drop" AllowDrop="True"
    HorizontalAlignment="Right" Margin="0,29,12,12" Width="215"
    ScrollViewer.VerticalScrollBarVisibility="Visible"/>
<TextBlock Height="21" Text="ListBox One" HorizontalAlignment="Left"
    Margin="12,2,0,0" VerticalAlignment="Top" Width="120" />
<TextBlock Height="21" Text="ListBox Two" HorizontalAlignment="Right"
    Margin="0,2,107,0" VerticalAlignment="Top" Width="120" />
</Grid>
</Window>

```

Now we would right code under the event handler ListBox_PreviewMouseLeftButtonDown.
 ListBox dragSource = null;

```

ListBox dragSource = null;

```

```

private void ListBox_PreviewMouseLeftButtonDown(object sender, MouseButtonEventArgs e)
{
    ListBox parent = (ListBox)sender;
    dragSource = parent;
    object data = GetDataFromListBox(dragSource, e.GetPosition(parent));

    if (data != null)
    {
        DragDrop.DoDragDrop(parent, data, DragDropEffects.Move);
    }
}

#region GetDataFromListBox(ListBox,Point)
private static object GetDataFromListBox(ListBox source, Point point)
{
    UIElement element = source.InputHitTest(point) as UIElement;
    if (element != null)
    {
        object data = DependencyProperty.UnsetValue;
        while (data == DependencyProperty.UnsetValue)
        {
            data = source.ItemContainerGenerator.ItemFromContainer(element);

            if (data == DependencyProperty.UnsetValue)
            {
                element = VisualTreeHelper.GetParent(element) as UIElement;
            }

            if (element == source)
            {
                return null;
            }
        }

        if (data != DependencyProperty.UnsetValue)
        {
            return data;
        }
    }
}

```

```

    }
}

return null;
}

#endregion

```

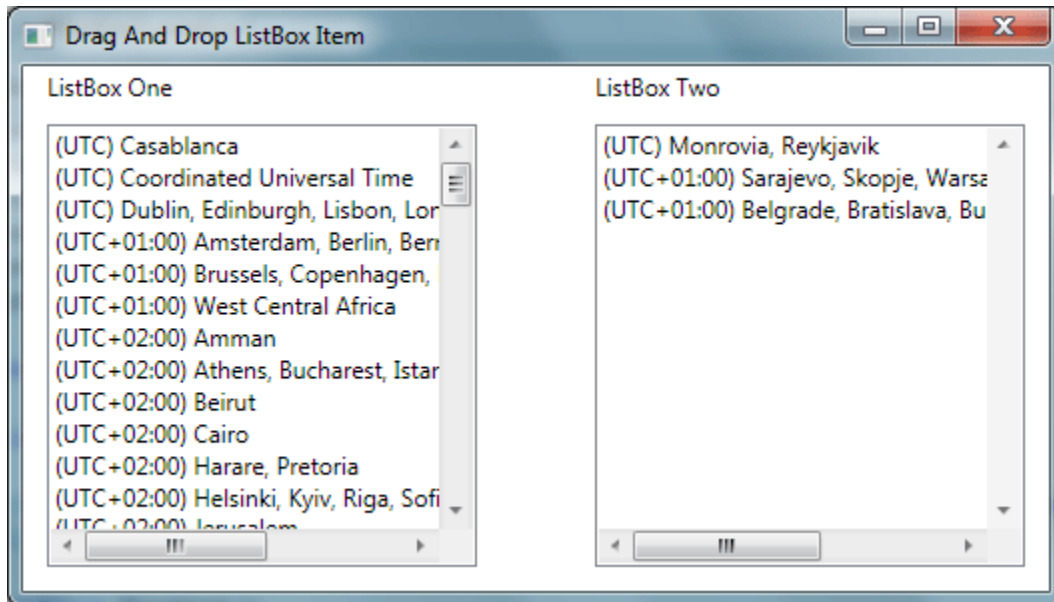
Now we would write code under ListBox_Drop event handler.

```

private void ListBox_Drop(object sender, DragEventArgs e)
{
    ListBox parent = (ListBox)sender;
    object data = e.Data.GetData(typeof(string));
    ((IList)dragSource.ItemsSource).Remove(data);
    parent.Items.Add(data);
}

```

That's it now run the application and Drag item from First ListBox and Drop into Second ListBox. You would achieve drag and drop functionality.



Hope this article helps.