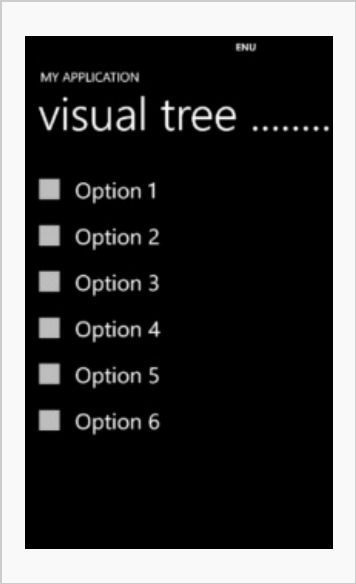
How to find a specific element inside a control using Visual tree in WP7

This article explains how to find a specific element inside a control using Visual tree in WP7. It uses VisualTreeHelper, a utility class which provides methods to traverse object relationships (along child object or parent object axes).

Introduction

`VisualTreeHelper` class is useful when we have to find a specific element/child within the parent control and perform some specific tasks such as modify/delete on the children. Parent controls can be any data bound control such as a `Grid`, `ListBox`, `StackPanel` with child controls including `TextBox`, `Buttons`, `TextBlocks`.

The simple application used in this code example displays a parent `ListBox` containing a number `CheckBox`. The example uses `VisualTreeHelper` to navigate the `ListBox` and count and list the items which have been "checked".



Sample Application Page

Textblock at the bottom provides the count and data associated with each selection

Getting Started

First create a Windows Phone application.

- Open Visual Studio and select **Windows Phone Application** from the installed templates
- Select Windows Phone 7.1 as the Target Version.

Defining the ListBox

The `ListBox` is defined in XAML, and populated using data binding from the C# code behind. The XAML definition is shown below:

```
<ListBox x:Name="my_list" Grid.Row="0">
    <ListBox.ItemTemplate>
        <DataTemplate>
            <StackPanel Orientation="Horizontal" >
                <CheckBox x:Name="cbx_state" Checked="cbx_state_Checked"
Unchecked="cbx_state_Unchecked" Tag="{Binding}"/>
                 <TextBlock x:Name="txt_string" Text="{Binding}"
VerticalAlignment="Center" FontSize="34" />
            </StackPanel>
         </DataTemplate>
```


(C) Copyright Nokia 2014. All rights reserved.

```
    </ListBox.ItemTemplate>
</ListBox>
```

The contents/appearance of the `ListBox` items are defined in terms of a Data Template; in this case each element is a `CheckBox` and a `TextBlock` horizontally aligned in a `StackPanel`. Note that the XAML above does not show the control used to display the results.

The `ListBox` is populated from C# as shown below by setting the content of the `my_list` declared in XAML:

```
my_list.ItemsSource = new List<string>() { "Option 1", "Option 2", "Option 3", "Option
4", "Option 5", "Option 6" };
```

Counting the checked Checkboxes

Below we define a generic method to search for a specified element/child:

```
//Variables to store the count of checked Checkboxes and their data
string option_selected = "";
int check_count = 0;

//SearchElement populates above variables for checkboxes in specified "targeted_control"
private void SearchElement(DependencyObject targeted_control)
  {
  var count = VisualTreeHelper.GetChildrenCount(targeted_control);   // targeted_control
is the listbox
  if (count > 0)
    {
    for (int i = 0; i < count; i++)
      {
      var child = VisualTreeHelper.GetChild(targeted_control, i);
      if (child is CheckBox) // specific/child control
        {
        CheckBox targeted_element = (CheckBox)child;
        if (targeted_element.IsChecked == true)
          {
          if (option_selected != "")
            {
            option_selected = option_selected + " , " + targeted_element.Tag;
            }
          else
            {
            // get the value associated with the "checked" checkbox
            option_selected = targeted_element.Tag.ToString();
            }
          // count the number of "Checked" checkboxes
          check_count = check_count + 1;
          return;
          }
        }
      else
        {
          SearchElement(child);
        }
      }
```

 (C) Copyright Nokia 2014. All rights reserved.

```
    }
  else
    {
    return;
    }
  }
```

The `SearchElement()` method is invoked each time when the `CheckBox` is checked or unchecked. In case the box is "checked":

```
private void cbx_state_Checked(object sender, RoutedEventArgs e)
    {
    check_count = 0;
    option_selected = "";
    SearchElement(my_list);
    txt_count.Text = " Total Selected Checkboxes : " + check_count.ToString()
         +Environment.NewLine + " Value Associated : " + option_selected;
    }
```

In case the box is "un-checked":

```
private void cbx_state_Unchecked(object sender, RoutedEventArgs e)
    {
    check_count = 0;
    option_selected = "";
    SearchElement(my_list);
    txt_count.Text = " Total Selected Checkboxes : " + check_count.ToString()
        +Environment.NewLine + " Value Associated : " + option_selected;
    }
```

Downloads

You can download sample project code from this file File:PhoneApp.zip.

 (C) Copyright Nokia 2014. All rights reserved.