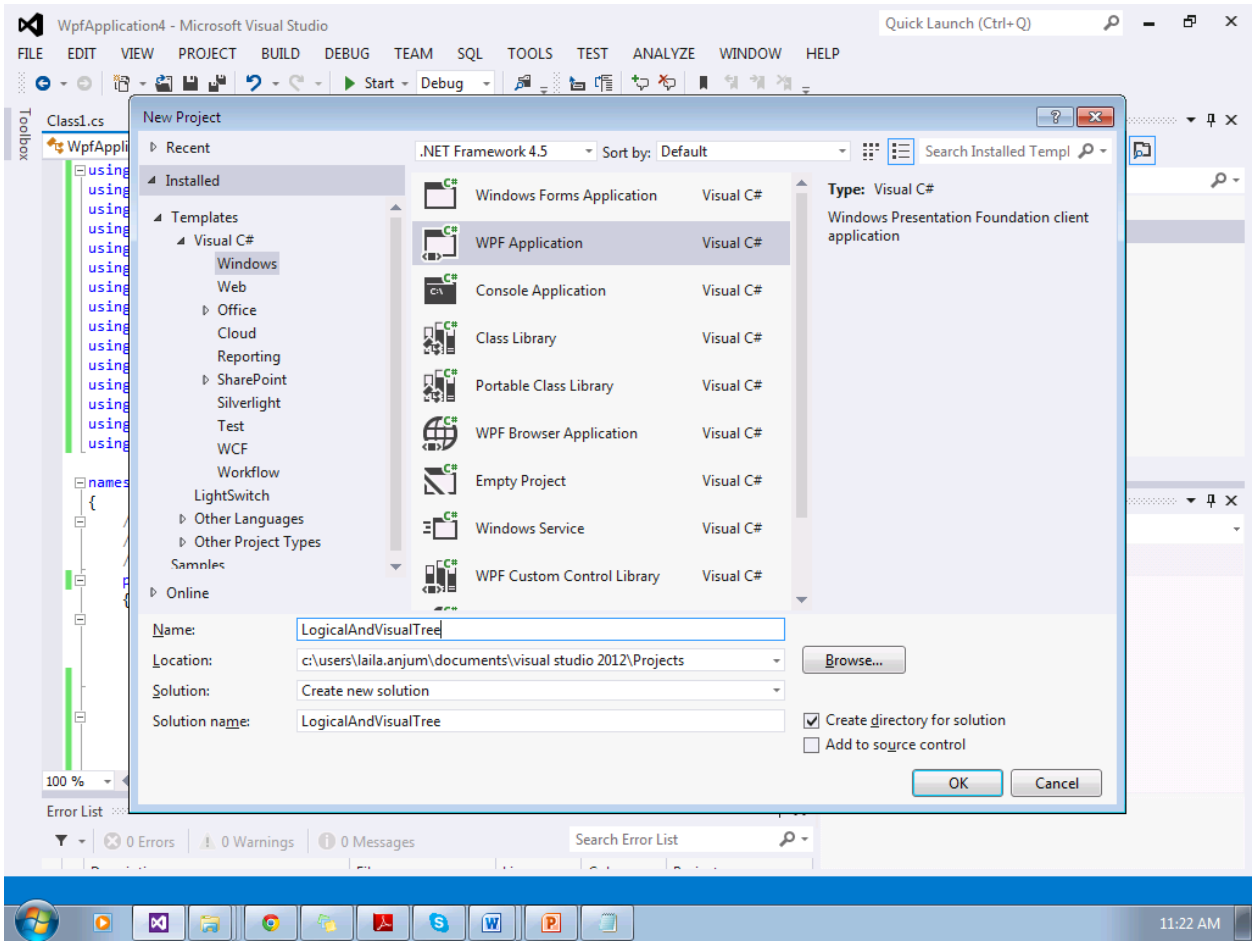
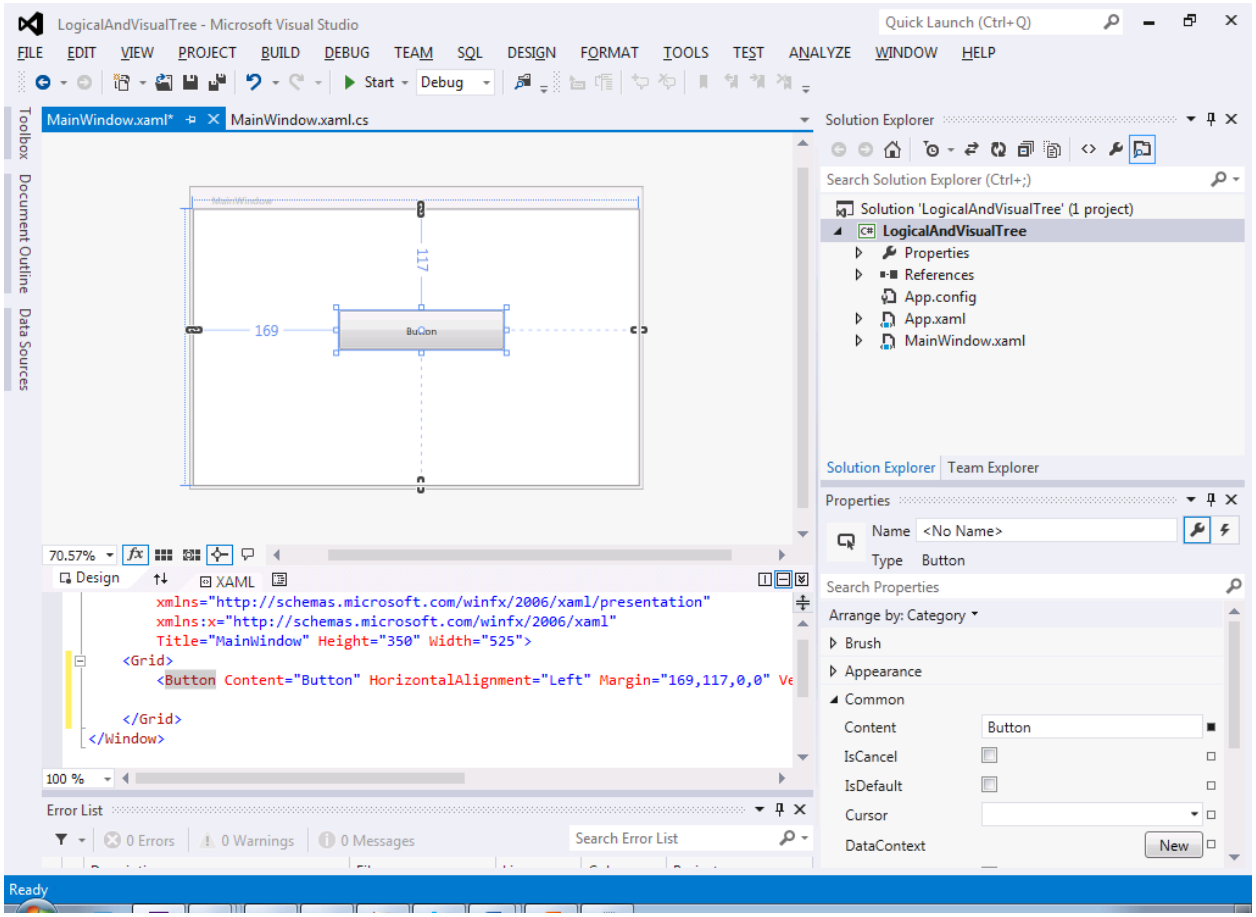
1. Create a new WPF Project “LogicalAndVisualTree”.

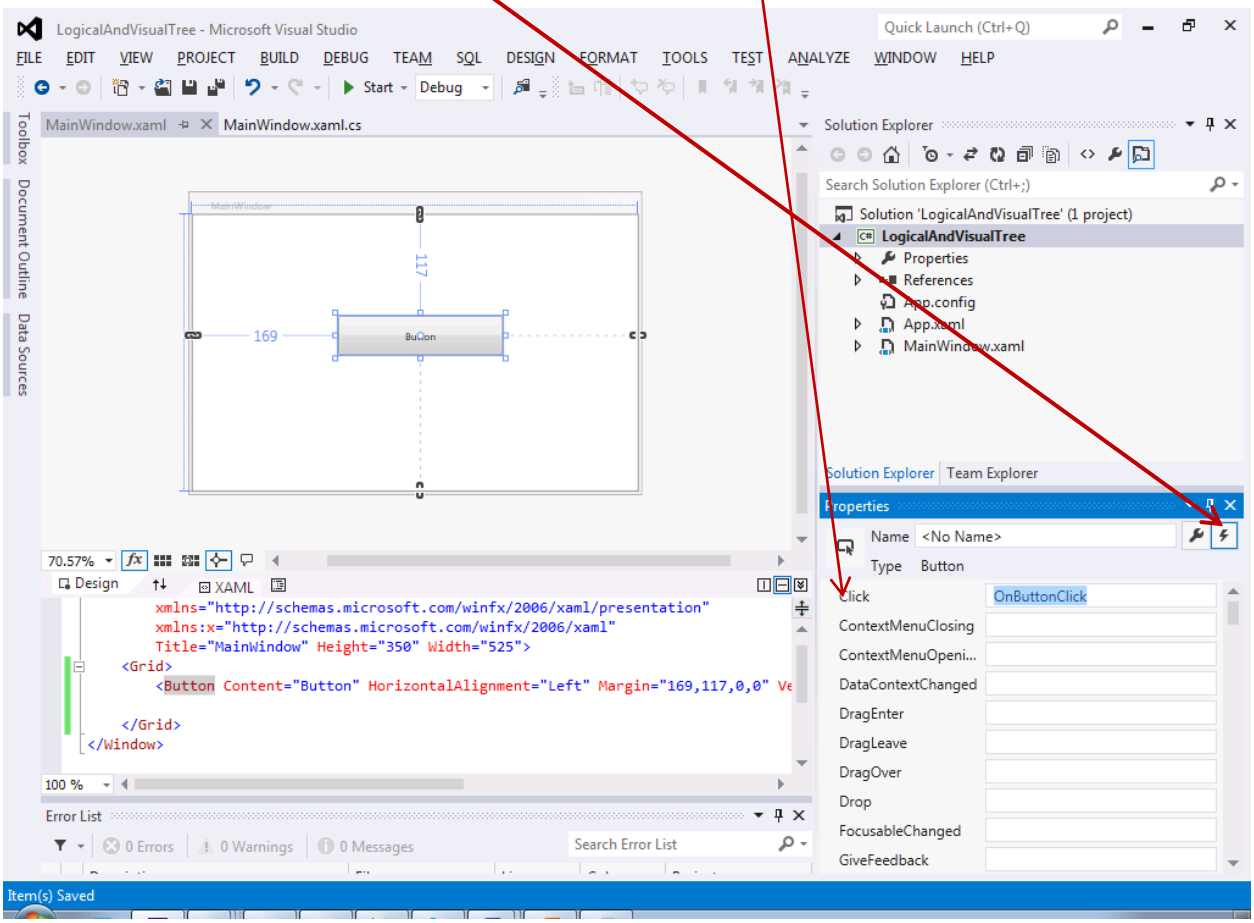


2. Add a button in Designer view.



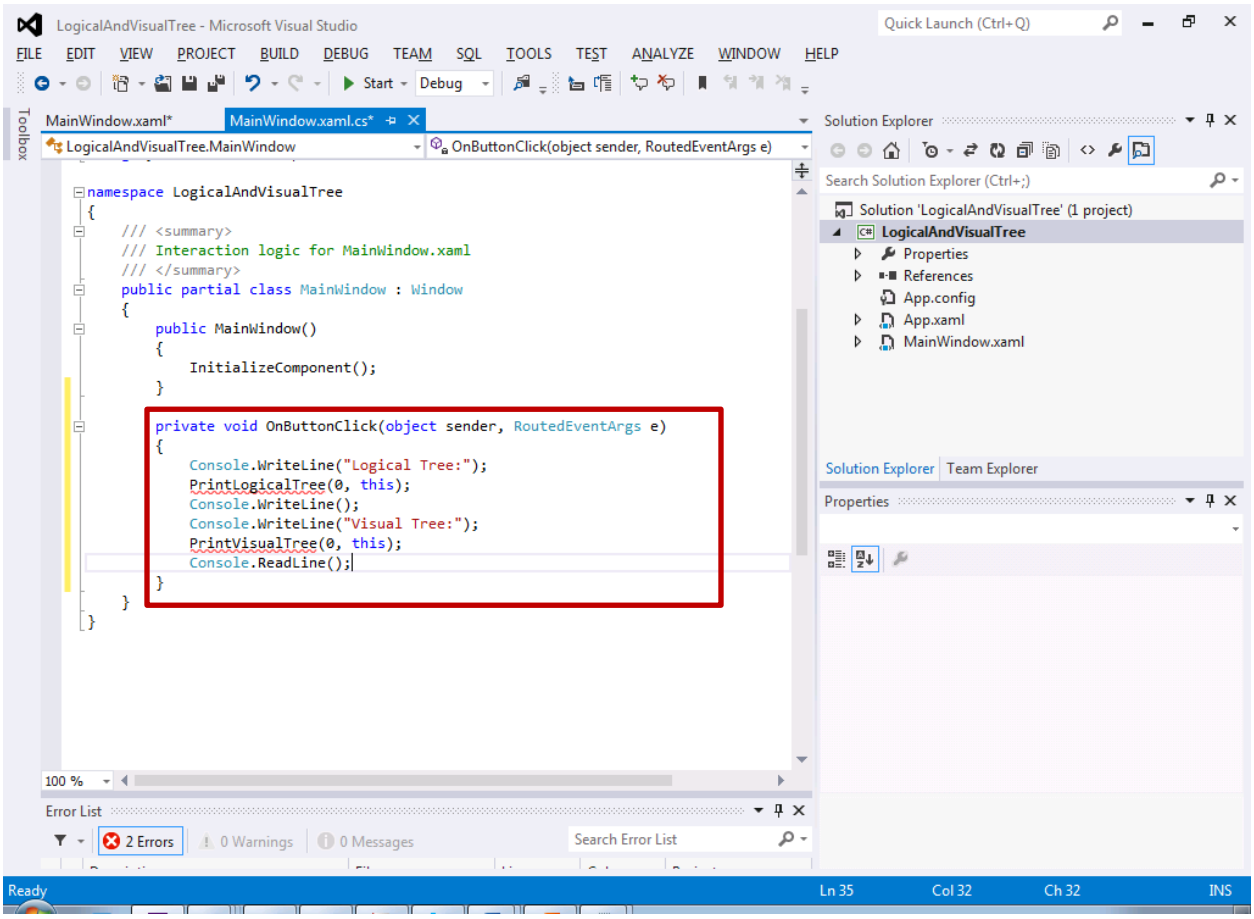
3. Add event to a button

- Click on this
- Add event “OnButtonClick” against “Click”
- Then press “Enter”



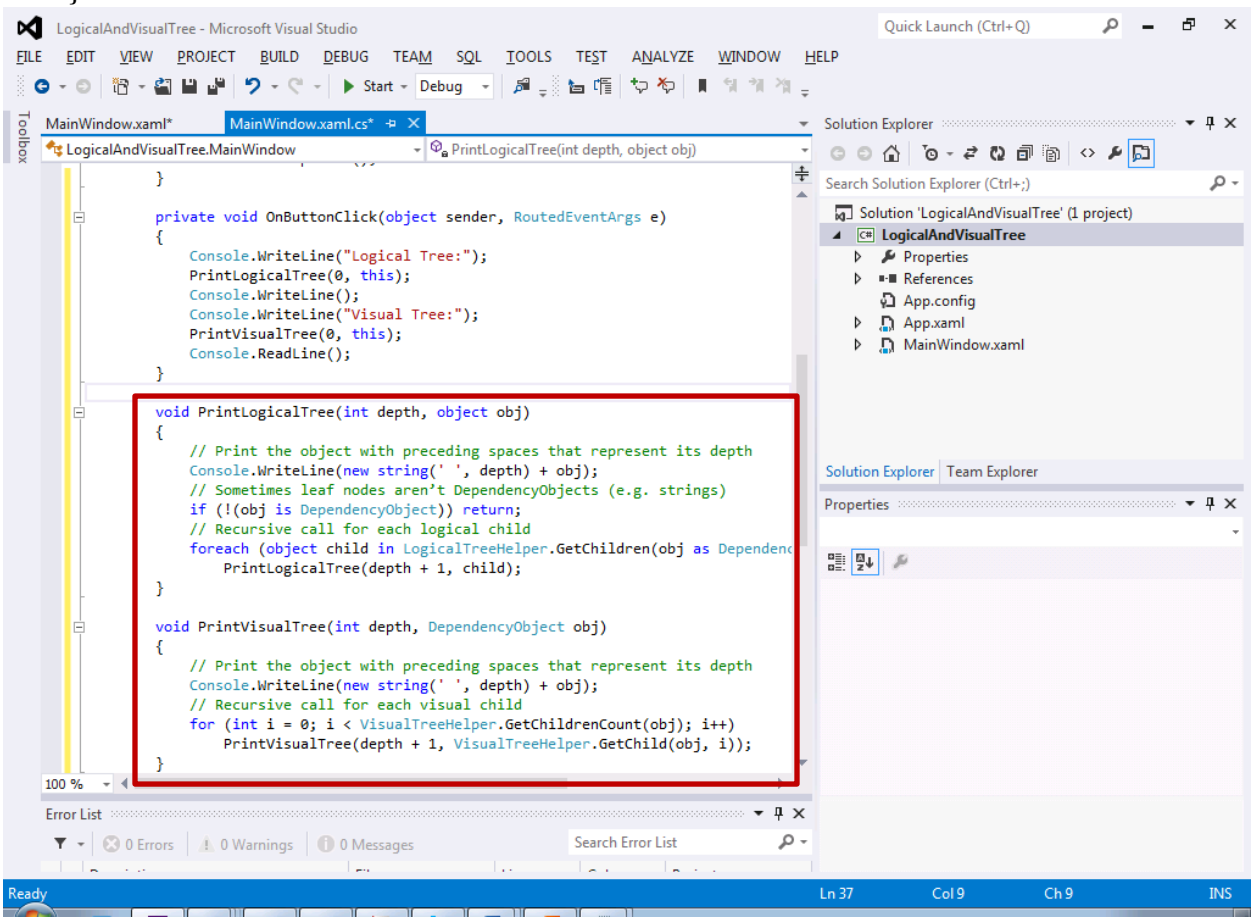
4. After pressing Enter, IDE will automatically take control to this page "MainWindow.xaml.cs" and a new function will be created with the name "OnButtonClick". This is the function where you can add functionality/events to your button. As the requirement of our tutorial is to Print Logical and Physical Tree when we click button, so for this add the following code lines in the highlighted portion:

```
Console.WriteLine("Logical Tree:");
PrintLogicalTree(0, this);
Console.WriteLine();
Console.WriteLine("Visual Tree:");
PrintVisualTree(0, this);
Console.ReadLine();
```

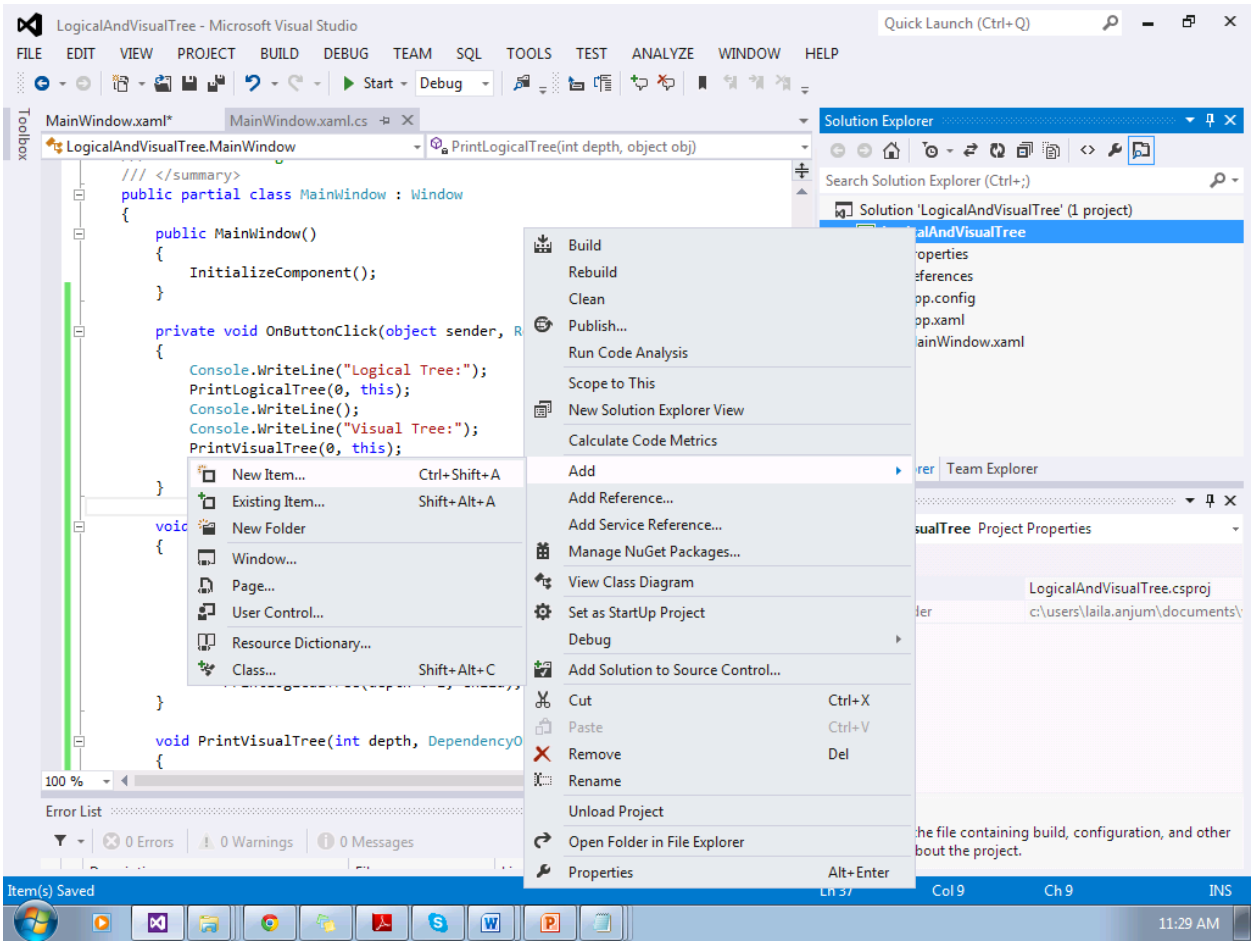


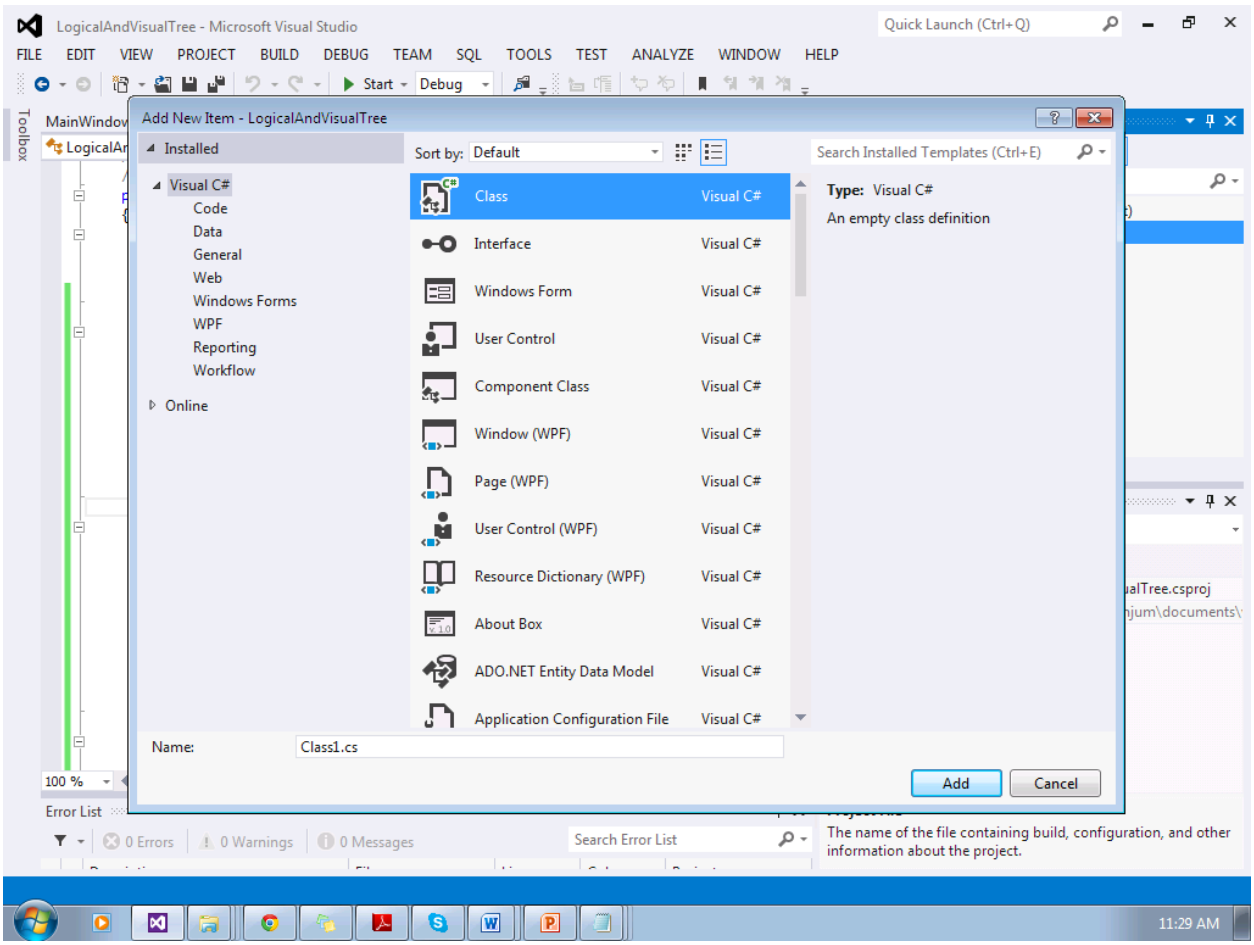
5. Add the following code after the "OnButtonClick" function. These functions walk through and print Logical and Physical trees.

```
void PrintLogicalTree(int depth, object obj)
    {
       // Print the object with preceding spaces that represent its depth
       Console.WriteLine(new string(' ', depth) + obj);
       // Sometimes leaf nodes aren't DependencyObjects (e.g. strings)
       if (!(obj is DependencyObject)) return;
       // Recursive call for each logical child
       foreach (object child in LogicalTreeHelper.GetChildren(obj as DependencyObject))
          PrintLogicalTree(depth + 1, child);
    }

  void PrintVisualTree(int depth, DependencyObject obj)
    {
       // Print the object with preceding spaces that represent its depth
       Console.WriteLine(new string(' ', depth) + obj);
       // Recursive call for each visual child
       for (int i = 0; i < VisualTreeHelper.GetChildrenCount(obj); i++)
          PrintVisualTree(depth + 1, VisualTreeHelper.GetChild(obj, i));
    }
```



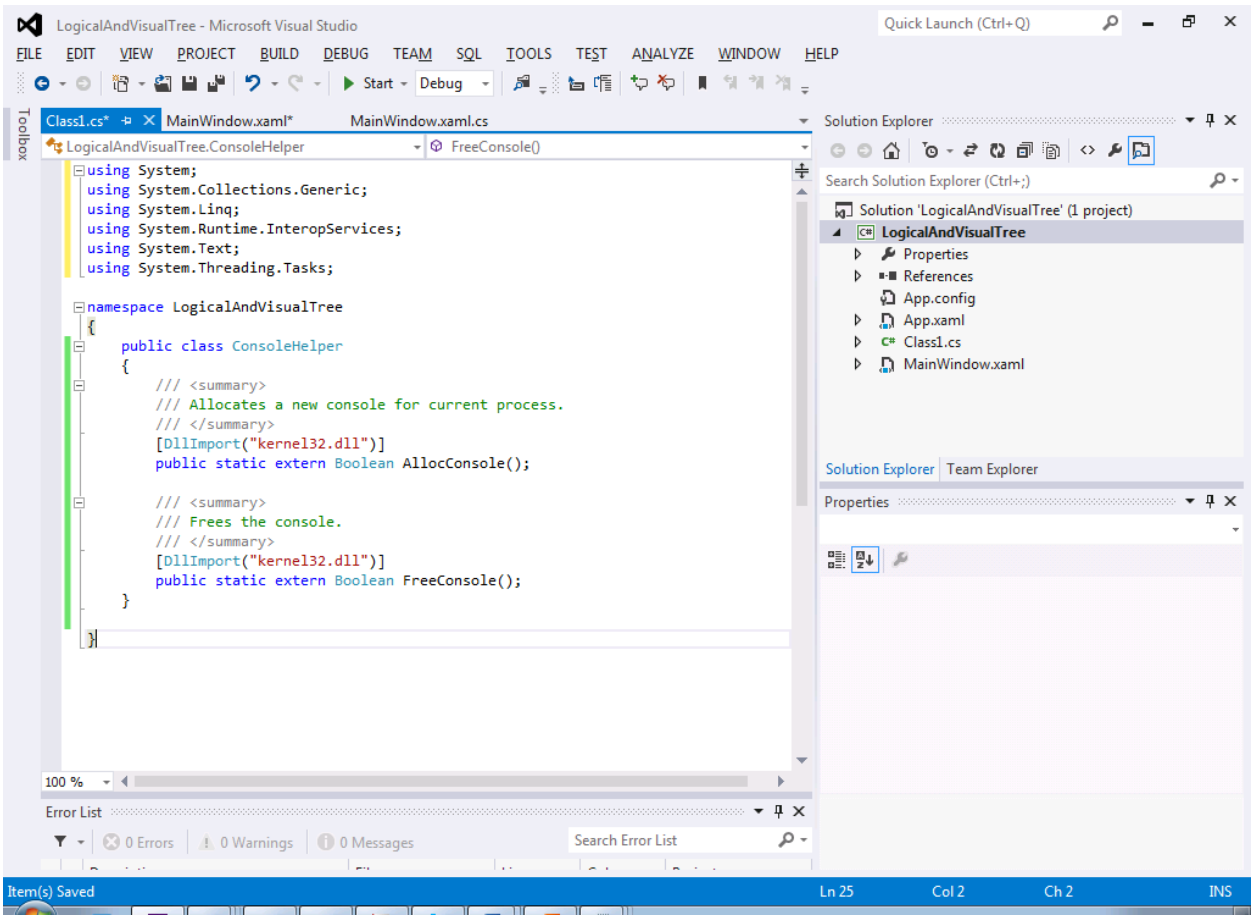
6. By default, WPF applications do not show the Console Output, so for this Add a new Class to the project by following the shown process.




LogicalAndVisualTree - Microsoft Visual Studio
Quick Launch (Ctrl+Q)
FILE EDIT VIEW PROJECT BUILD DEBUG TEAM SQL TOOLS TEST ANALYZE WINDOW HELP
Start
Debug
Toolbox
MainWindow
LogicalAr
Add New Item - LogicalAndVisualTree
Installed
Visual C#
Code
Data
General
Web
Windows Forms
WPF
Reporting
Workflow
Online
Sort by: Default
Search Installed Templates (Ctrl+E)
Class Visual C#
Interface Visual C#
Windows Form Visual C#
User Control Visual C#
Component Class Visual C#
Window (WPF) Visual C#
Page (WPF) Visual C#
User Control (WPF) Visual C#
Resource Dictionary (WPF) Visual C#
About Box Visual C#
ADO.NET Entity Data Model Visual C#
Application Configuration File Visual C#
Type: Visual C#
An empty class definition
Name: Class1.cs
Add
Cancel
100 %
Error List
0 Errors
0 Warnings
0 Messages
Search Error List
The name of the file containing build, configuration, and other information about the project.
11:29 AM

7. Now copy paste the following code in the newly created file.

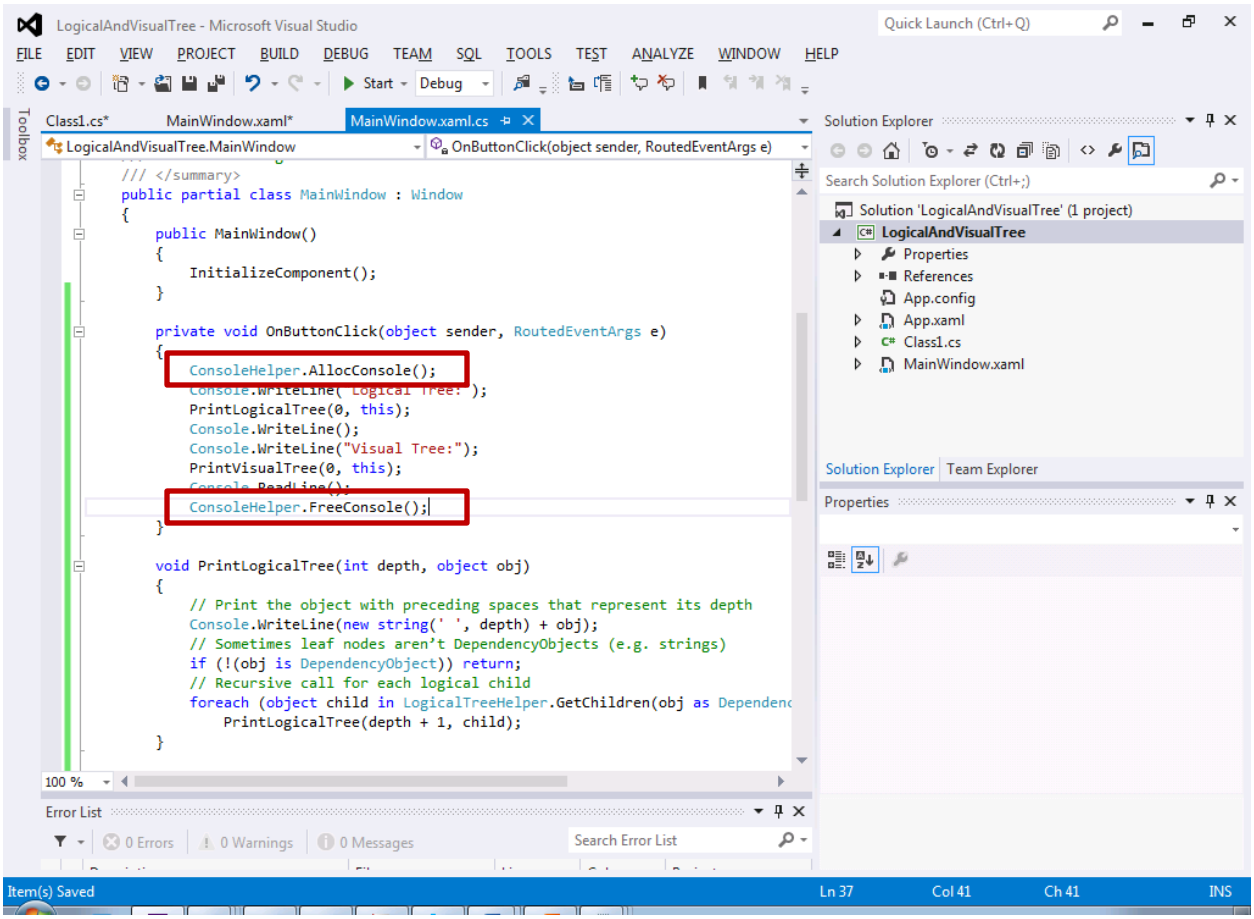
```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Runtime.InteropServices;
using System.Text;
using System.Threading.Tasks;

namespace LogicalAndVisualTree
{
   public class ConsoleHelper
   {
      /// <summary>
      /// Allocates a new console for current process.
      /// </summary>
      [DllImport("kernel32.dll")]
      public static extern Boolean AllocConsole();

      /// <summary>
      /// Frees the console.
      /// </summary>
      [DllImport("kernel32.dll")]
      public static extern Boolean FreeConsole();
   }

}
```


LogicalAndVisualTree - Microsoft Visual Studio
Quick Launch (Ctrl+Q)
FILE EDIT VIEW PROJECT BUILD DEBUG TEAM SQL TOOLS TEST ANALYZE WINDOW HELP
Start Debug
Toolbox
Class1.cs* MainWindow.xaml* MainWindow.xaml.cs
LogicalAndVisualTree.ConsoleHelper
FreeConsole()
using System;
using System.Collections.Generic;
using System.Linq;
using System.Runtime.InteropServices;
using System.Text;
using System.Threading.Tasks;
namespace LogicalAndVisualTree
{
public class ConsoleHelper
{
/// <summary>
/// Allocates a new console for current process.
/// </summary>
[DllImport("kernel32.dll")]
public static extern Boolean AllocConsole();
/// <summary>
/// Frees the console.
/// </summary>
[DllImport("kernel32.dll")]
public static extern Boolean FreeConsole();
}
}
100 %
Solution Explorer
Search Solution Explorer (Ctrl+;)
Solution 'LogicalAndVisualTree' (1 project)
LogicalAndVisualTree
Properties
References
App.config
App.xaml
Class1.cs
MainWindow.xaml
Solution Explorer Team Explorer
Properties
Error List
0 Errors 0 Warnings 0 Messages
Search Error List
Item(s) Saved
Ln 25 Col 2 Ch 2 INS

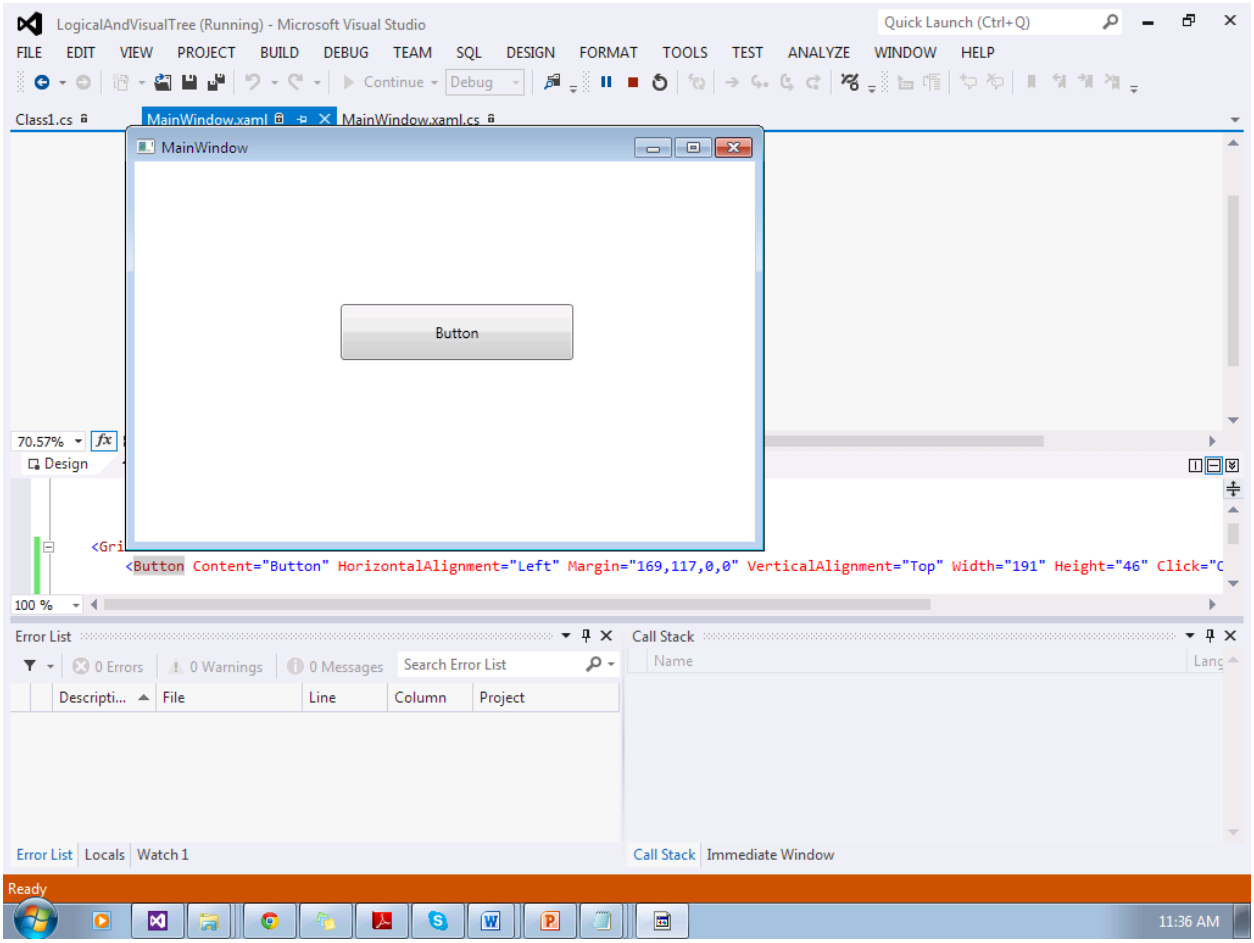
8. Now add the following code lines in the specified positions:

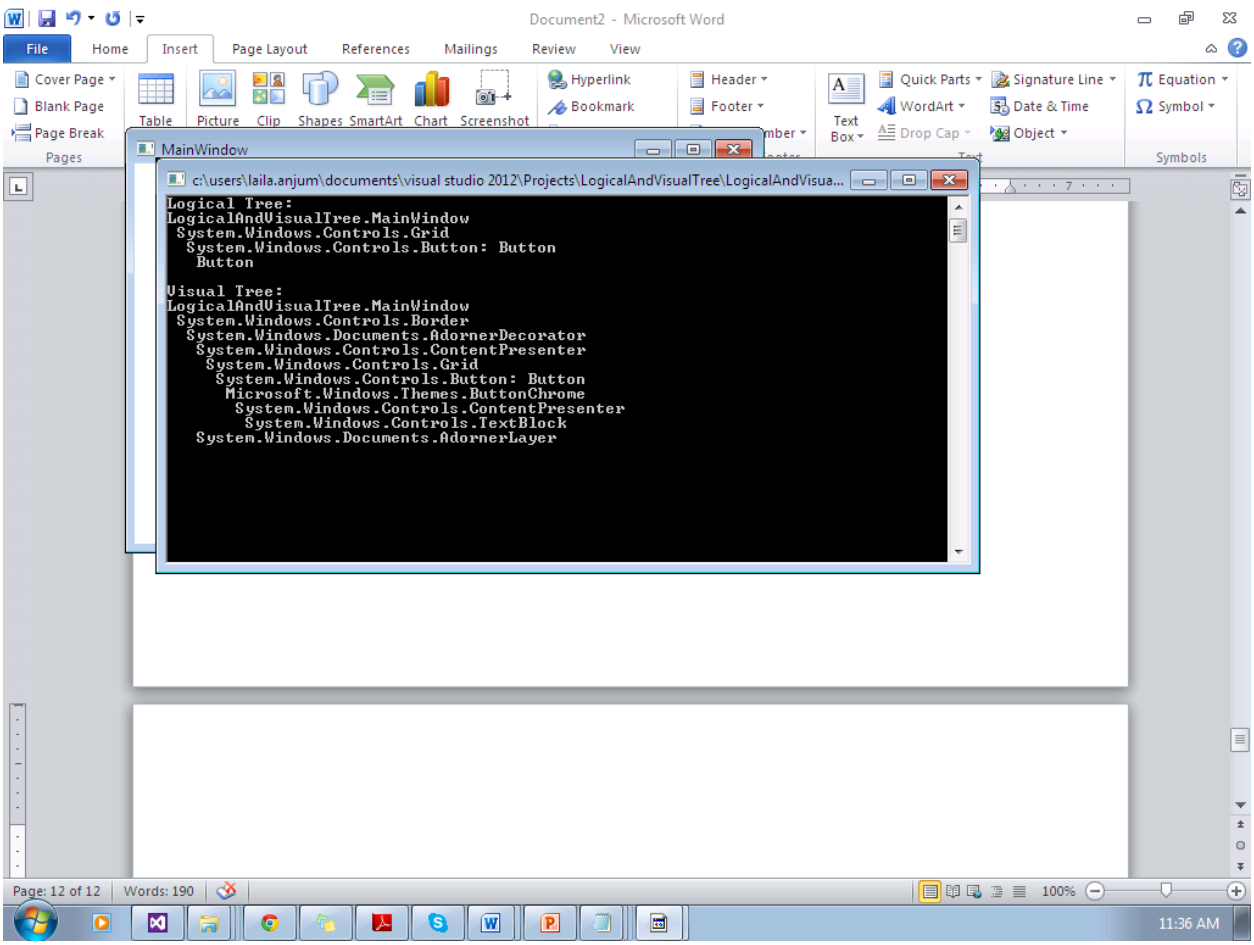
ConsoleHelper.AllocConsole();

ConsoleHelper.FreeConsole();



9. Now Press “F5” to execute the Program. Click on Button, it will show the Console Output, displaying the Logical and Physical Trees.





In the same way, you can view Logical and Physical Tree of any control.