XSLT transform GUI with SharpDevelop IDE*

Hannes Hirzel

This work is produced by The Connexions Project and licensed under the
Creative Commons Attribution License †

Abstract

It is shown how to do program a simple GUI with SharpDevelop to do XSLT transforms.

1 Introduction

In this module a simple GUI for doing XSLT transforms is created. The IDE used is SharpDevelop[1] , the language is C-Sharp. In another module the code is converted to VB.net, see links section.

The module is in a draft stage at the moment. It probably gives enough details for some readers, for others there is not enough information to recreate the example.

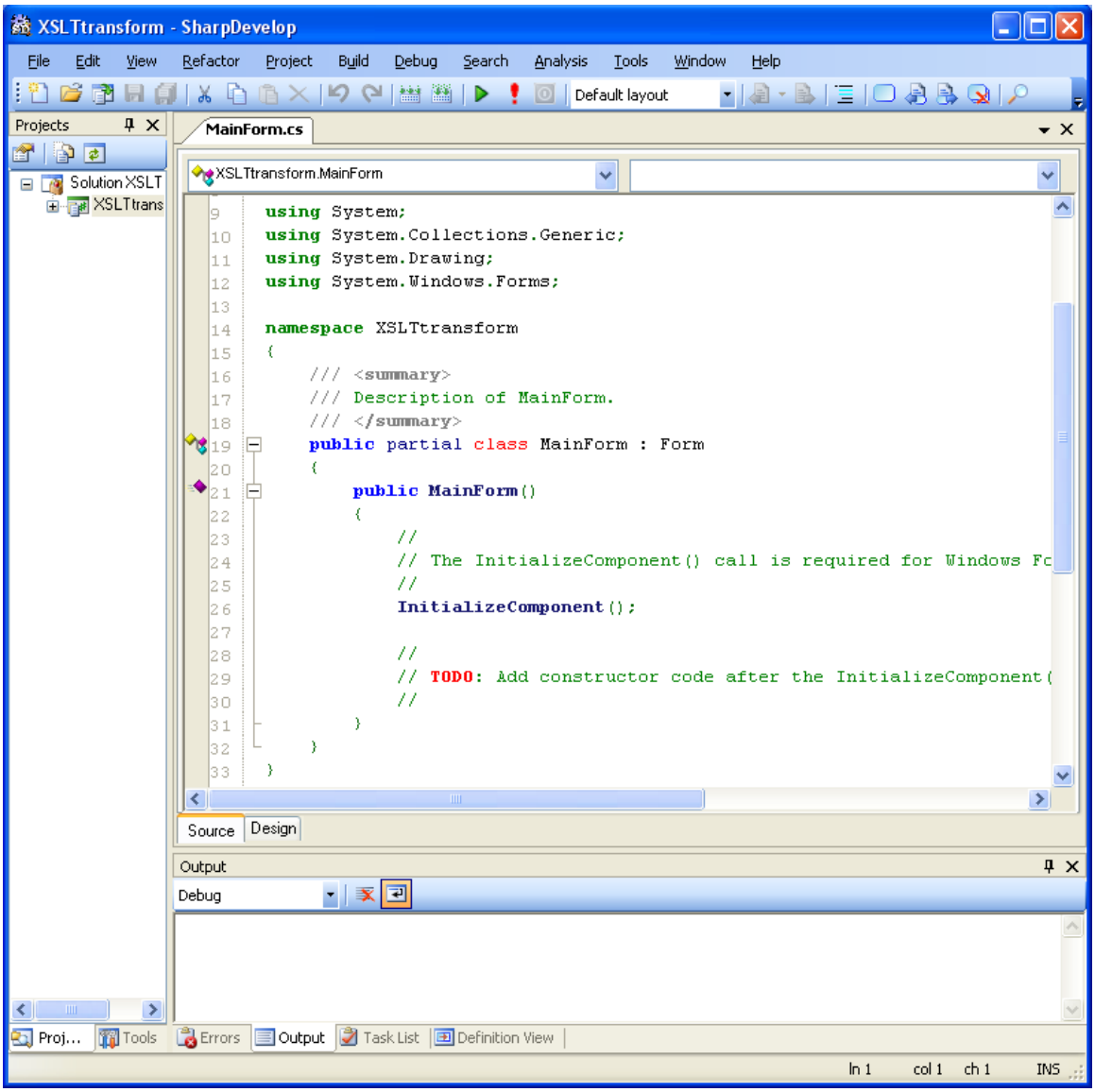
2 Create the XSLT transform GUI

Start the IDE. Choose 'File' / 'New Solution', then 'Windows Application'. Then give a name and click 'Create'. You should now have what the screen shot below shows.

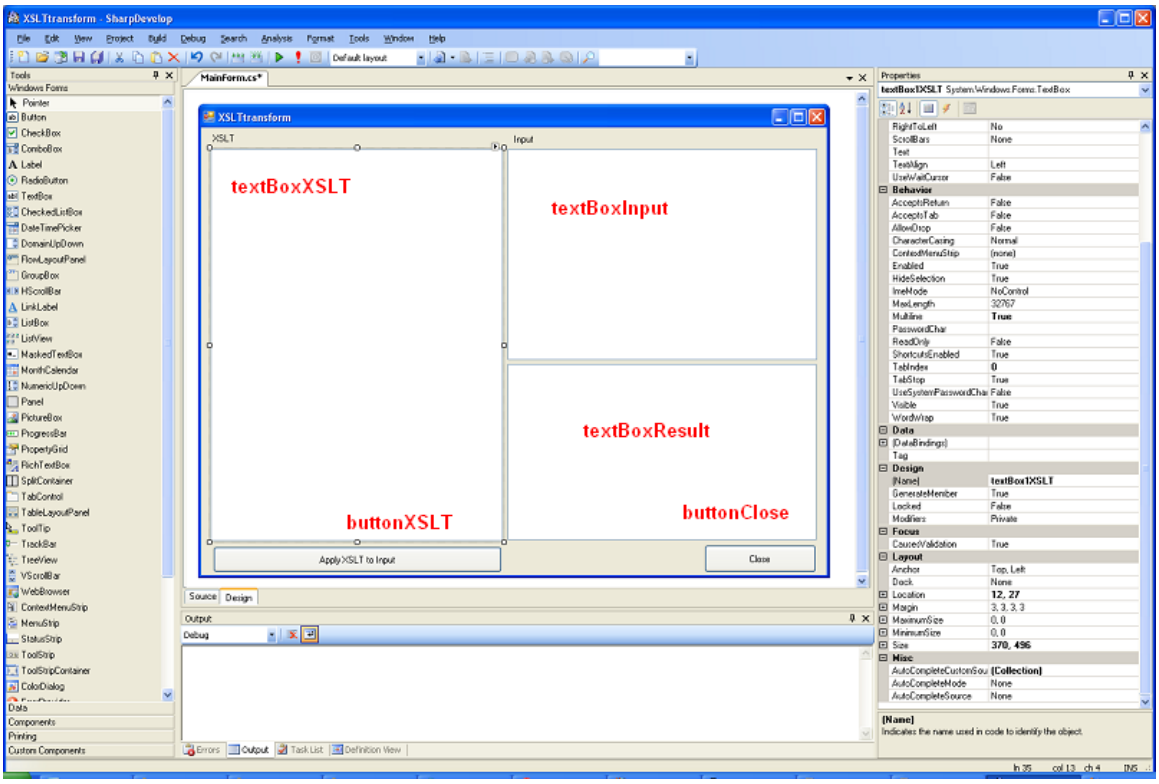
*Version 1.5: Oct 8, 2012 3:38 am -0500

†http://creativecommons.org/licenses/by/3.0/

[1]http://www.icsharpcode.net/OpenSource/SD/Default.aspx



Click on 'Design' on the tab below the MainForm, if necessary choose 'Properties' from the 'View' menu to get a properties window.



Create the following GUI elements

- textBoxXSLT
- textBoxInputXML
- textBoxResult
- buttonXSLT
- buttonClose

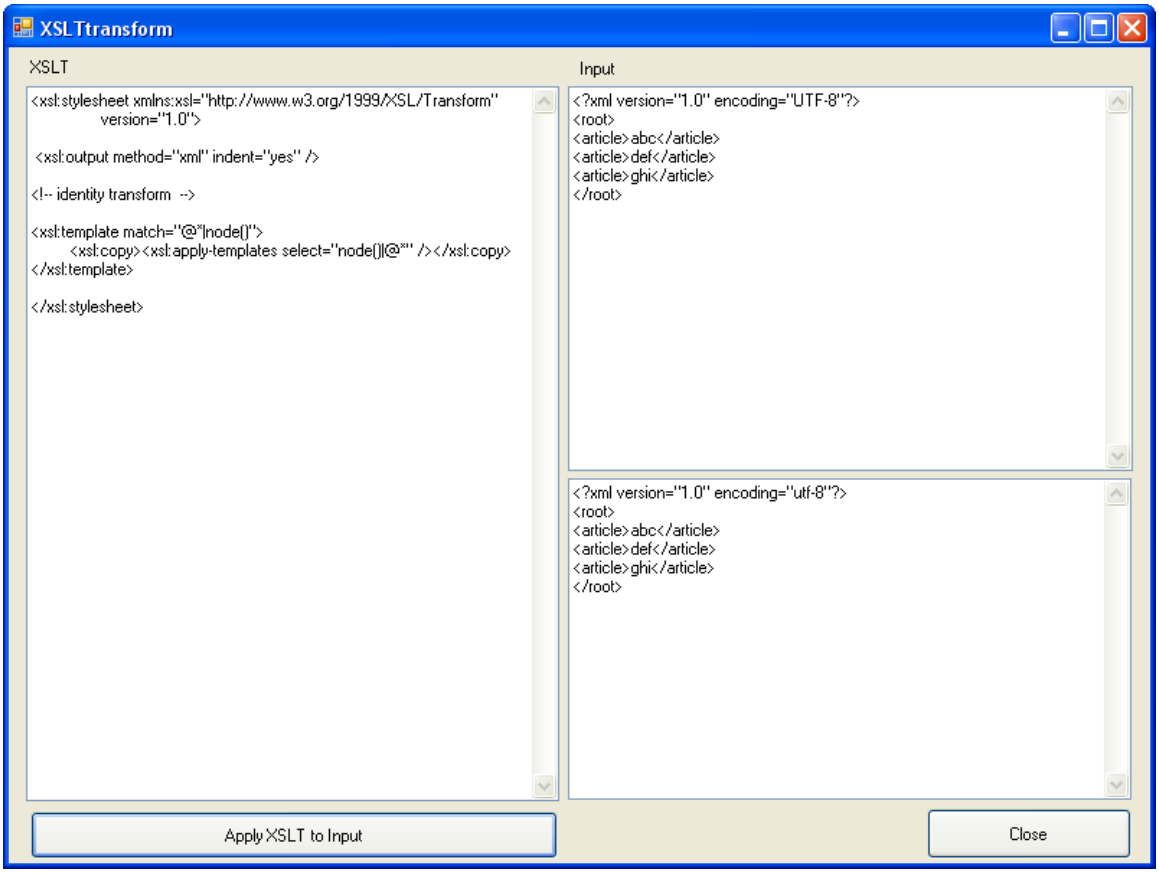
For the text boxes set the property `multiline` to `true`.

You might also want to set `maxlength` of the text boxes to a higher number, e.g. 1000000.

Then adapt the code that the text boxes are initialized and the XSLT button does an XSLT transform as shown below. (MORE EXPLANATIONS NEEDED)

```
  /*
 * Created by SharpDevelop.
 * User: hirzel
 * Date: 09/05/2012
 * Time: 06:03
 *
 * To change this template use Tools | Options | Coding | Edit Standard Headers.
 */
using System;
using System.Collections.Generic;
using System.Drawing;
using System.Windows.Forms;
using System.IO;
using System.Xml.Xsl;
```

```
namespace XSLTtransform
{
/// <summary>
/// Description of MainForm.
/// </summary>
public partial class MainForm : Form
{
public MainForm()
{
//
// The InitializeComponent() call is required for Windows Forms designer support.
//
InitializeComponent();

//
// TODO: Add constructor code after the InitializeComponent() call.
//
if (File.Exists("transform.xsl")) {
StreamReader sr = new StreamReader("transform.xsl");
this.textBoxXSLT.Text = sr.ReadToEnd();
sr.Close();
};
buttonXSLT.Select();
}




void ButtonXSLTClick(object sender, EventArgs e)
{
// Console.Write(Path.GetTempPath());
StreamWriter sw = new StreamWriter("input.xml");
sw.Write(textBoxInputXML.Text);
sw.Close();

sw = new StreamWriter("transform.xsl");
sw.Write(textBoxXSLT.Text);
sw.Close();

// Create the XsltSettings object with script enabled.

XsltSettings settings = new XsltSettings(false,true);

// Execute the transform.
try {
  XslCompiledTransform xslt = new XslCompiledTransform();
xslt.Load("transform.xsl", settings, new System.Xml.XmlUrlResolver());
xslt.Transform("input.xml", "output.xml");

StreamReader sr = new StreamReader("output.xml");
textBoxResult.Text = sr.ReadToEnd();
```

```
sr.Close();
} catch (Exception ex) {
MessageBox.Show(ex.Message);

}
}

void Button2Click(object sender, EventArgs e)
{
this.Close();
}
}
}
```

3 The working program