

Forum:[Hack Forums](#)

Thread: [\[Tut\]Make your own simple C# Compiler](#)

BaussHacker - May 4, 2011, 8:47 am

First of all create a new form.

Then we need to set it up simple.

First thing we do is making a panel.

Drag the panel to the form and set the dock to "Top".

Drag a toolstrip into the panel.

Now drag another panel into the form and set the dock to "Right".

And then a last panel into our form. The dock of that should be "Fill".

It should look something like this now:

Spoiler ([Click to View](#))

[IMG]<http://i590.photobucket.com/albums/ss347/PhoeNix4Real/cs1.png>[/IMG]

The next thing we do is dragging a tabcontrol into the panel3.

Delete the 2tabs it have and make the dock "Fill".

Now in the panel2 put a label, a textbox, 2 radiobuttons and a listbox.

The label should have the text "Assembly name".

The first radiobutton should have the text "Class Library" and it should be checked. The second should have the text "Console application".

The listbox should have dock to "Bottom".

Now it should look something like this:

Spoiler ([Click to View](#))

[IMG]<http://i590.photobucket.com/albums/ss347/PhoeNix4Real/cs2.png>[/IMG]

The next thing we do is adding a 2 dropdowns to our toolstrip.

One called "File" and one called "Project".

Set the File displaystyle to "Text".

Make sure ShowDropDownArrow is set to "false".

Following dropdownitems should be added.

Code New File

Open File

Save File

Set the Project displaystyle to "Text".

Make sure ShowDropDownArrow is set to "false".

Following dropdownitems should be added.

Code Compile And Run

Compile Only

Add Reference

It should all look like this now:

Spoiler ([Click to View](#))

[IMG]http://i590.photobucket.com/albums/ss347/PhoeNix4Real/cs3.png[/IMG]

Now the base GUI is done and we will start on the coding.

First thing we do is adding the "New File".

Double click on it and you should get this code:

```
Code private void newFileToolStripMenuItem_Click(object sender, EventArgs e)
{
```

```
}
```

Replace it with:

```
Code private void newFileToolStripMenuItem_Click(object sender, EventArgs e)
{
```

```
   TabPage page = new TabPage();
```

```
    page.Name = "New File";
```

```
    page.Text = "Unnamed.cs";
```

```
    RichTextBox rtb = new RichTextBox();
```

```
    rtb.Name = "richTextBox1";
```

```
    rtb.Size = new System.Drawing.Size(100, 96);
```

```
    rtb.TabIndex = 5;
```

```
    rtb.Dock = DockStyle.Fill;
```

```
    rtb.BorderStyle = BorderStyle.None;
```

```
    rtb.Font = new System.Drawing.Font("Consolas", 9.75F, System.Drawing.FontStyle.Regular,
System.Drawing.GraphicsUnit.Point, ((byte)0));
```

```
    rtb.Text = @"using System;
```

```
namespace Application
```

```
{
```

```
    class Program
```

```
    {
```

```
        static void Main()
```

```
        {
```

```
            Console.WriteLine("Hello World");
```

```
        }
```

```
    }
```

```
};
```

```
    page.Controls.Add(rtb);
```

```
    tabControl1.Controls.Add(page);
```

```
}
```

Now we can create new files.

Now click on the Open File.

You should get this code:

```
Code private void openFileToolStripMenuItem_Click(object sender, EventArgs e)
{
```

```
}
```

Replace it with:

```
Code private void openFileToolStripMenuItem_Click(object sender, EventArgs e)
{
```

```
    OpenFileDialog opn = new OpenFileDialog();
```

```
    opn.Title = "Open File";
```

```
    opn.Filter = "CSharp Coding Files|.cs";
```

```
opn.Multiselect = false;
```

```
if (opn.ShowDialog() == System.Windows.Forms.DialogResult.OK)
{
    string Name = System.IO.Path.GetFileName(opn.FileName);
    TabPage page = new TabPage();
    page.Name = "New File";
    page.Text = Name;
```

```
    RichTextBox rtb = new RichTextBox();
    rtb.Name = opn.FileName;
    rtb.Size = new System.Drawing.Size(100, 96);
    rtb.TabIndex = 5;
    rtb.Dock = DockStyle.Fill;
    rtb.BorderStyle = BorderStyle.None;
    rtb.Font = new System.Drawing.Font("Consolas", 9.75F, System.Drawing.FontStyle.Regular,
    System.Drawing.GraphicsUnit.Point, ((byte)0));
```

```
    rtb.Text = System.IO.File.ReadAllText(opn.FileName);
    page.Controls.Add(rtb);
    tabControl1.Controls.Add(page);
}
```

Now click the Save File and you should get this code:

```
Code private void saveFileToolStripMenuItem_Click(object sender, EventArgs e)
{
}
```

Replace it with:

```
Code private void saveFileToolStripMenuItem_Click(object sender, EventArgs e)
{
    SaveFileDialog save = new SaveFileDialog();
    save.Title = "Save File";
    save.Filter = "CSharp Coding Files|*.cs";
```

```
    if (save.ShowDialog() == System.Windows.Forms.DialogResult.OK)
    {
        if (save.CheckFileExists)
        {
            DialogResult dlg = MessageBox.Show("Are you sure you want to overwrite this file?", "Warning",
            MessageBoxButtons.YesNo, MessageBoxIcon.Warning);
            if (dlg == System.Windows.Forms.DialogResult.No || dlg ==
            System.Windows.Forms.DialogResult.Cancel)
                return;
        }
    }
```

```
    FileStream fs;
    try
    {
        fs = new FileStream(save.FileName, FileMode.Truncate);
    }
    catch
    {
        fs = new FileStream(save.FileName, FileMode.CreateNew);
    }
```

```

}
StreamWriter sw = new StreamWriter(fs);
foreach (Control ctrl in tabControl1.Controls)
if (ctrl is TabPage)
{
foreach (Control Ctrl in ctrl.Controls)
if (Ctrl is RichTextBox)
{
sw.WriteLine(Ctrl.Text);
}
}
sw.Close();
fs.Close();
tabControl1.SelectedTab.Text = System.IO.Path.GetFileName(save.FileName);
}
}

```

Now go to the dropdown Project and click on Add Reference.

You should get this code:

```

Code private void addReferenceToolStripMenuItem_Click(object sender, EventArgs e)
{

```

```

}

```

Replace it with:

```

Code private void addReferenceToolStripMenuItem_Click(object sender, EventArgs e)
{

```

```

    OpenFileDialog opn = new OpenFileDialog();
    opn.Title = "Add Reference";
    opn.Filter = "Libraries|*.dll";
    opn.Multiselect = false;
    if (opn.ShowDialog() == DialogResult.OK)
    {
        if (!Directory.Exists(Environment.CurrentDirectory + @"\References"))
        {
            DirectoryInfo dir = new DirectoryInfo(Environment.CurrentDirectory + @"\References");
            dir.Create();
        }
        string[] FILE = opn.FileName.Split(new char[] { '\\' });
        File.Copy(opn.FileName, Environment.CurrentDirectory + @"\References\" + FILE[FILE.Length -
1], true);
        listBox1.Items.Add(FILE[FILE.Length - 1]);
    }
}

```

Now make a new class and call it Compiler.

Replace it all with this:

```

Code using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using Microsoft.CSharp;
using System.CodeDom.Compiler;

```

```

namespace CS_Compiler
{

```

```

public class Compiler
{
public Compiler(string ApplicationName, bool EXE, string Code, string[] References, bool run)
{
    ApplicationName = ApplicationName.Split('.')[0]; //To make sure we won't use a wrong extension.
    if (EXE)
        ApplicationName += ".exe";
    else
        ApplicationName += ".dll";

    CSharpCodeProvider codeProvider = new CSharpCodeProvider();
    ICodeCompiler icc = codeProvider.CreateCompiler();
    System.CodeDom.Compiler.CompilerParameters parameters = new CompilerParameters();
    parameters.GenerateExecutable = EXE;
    parameters.OutputAssembly = ApplicationName;
    if (References != null)
    {
        for (int i = 0; i < References.Length; i++)
        {
            parameters.ReferencedAssemblies.Add(References[i]);
        }
    }
    CompilerResults results = icc.CompileAssemblyFromSource(parameters, Code);
    //Use results.Errors to get all the errors compiling.

    if (run && EXE)
        System.Diagnostics.Process.Start(ApplicationName);
    }
}

```

If you want to get all the errors throwing at the compilation, then use results.Errors. You can make a textbox at the bottom of your IDE which can show the errors. The same goes for output.

Now we just need to make the compilation done.

Click on the Compile And Run.

You should get this code:

```

Code private void compileAndRunToolStripMenuItem_Click(object sender, EventArgs e)
{
}

```

Replace it with:

```

Code private void compileAndRunToolStripMenuItem_Click(object sender, EventArgs e)
{
    bool exe = false;
    bool run = true;

    if (radioButton2.Checked)
        exe = true;
    else
    {
        run = false;
        exe = false;
    }
}

```

```
string[] References = new string[listBox1.Items.Count];
for (int i = 0; i < References.Length; i++)
References[i] = @"References\" + listBox1.Items[i].ToString();
```

```
if (!Directory.Exists(Environment.CurrentDirectory + @"\Builds"))
{
DirectoryInfo dir = new DirectoryInfo(Environment.CurrentDirectory + @"\Builds");
dir.Create();
}
```

```
string Code = "";
foreach (Control ctrl in tabControl1.Controls)
if (ctrl is TabPage)
{
foreach (Control Ctrl in ctrl.Controls)
if (Ctrl is RichTextBox)
{
Code = Ctrl.Text;
}
}
}
```

```
Compiler compiler = new Compiler(@"Builds\" + textBox1.Text, exe, Code, References, run);
}
```

Now click on the Compile Only and you should get this code:

```
Code private void compileOnlyToolStripMenuItem_Click(object sender, EventArgs e)
{
}
}
```

Replace it with:

```
Code private void compileOnlyToolStripMenuItem_Click(object sender, EventArgs e)
{
bool exe = false;

if (radioButton2.Checked)
exe = true;
else
exe = false;
```

```
string[] References = new string[listBox1.Items.Count];
for (int i = 0; i < References.Length; i++)
References[i] = @"References\" + listBox1.Items[i].ToString();
```

```
if (!Directory.Exists(Environment.CurrentDirectory + @"\Builds"))
{
DirectoryInfo dir = new DirectoryInfo(Environment.CurrentDirectory + @"\Builds");
dir.Create();
}
```

```
string Code = "";
foreach (Control ctrl in tabControl1.Controls)
if (ctrl is TabPage)
{
```

```
foreach (Control Ctrl in ctrl.Controls)
if (Ctrl is RichTextBox)
{
Code = Ctrl.Text;
}
}
```

```
Compiler compiler = new Compiler(@"Builds\" + textBox1.Text, exe, Code, References, false);
}
```

Now you have your own simple C# Compiler.
I hope you liked it and find it useful.
?

MaxPain - May 4, 2011, 9:53 am

Awesome Idea And Great Job As Always BauussHacker

BauussHacker - May 4, 2011, 1:09 pm

Thank you for the feedback :)

#Edit

If anybody needs help or anything with adding new features etc. Let me know.

Logic_Bomber - May 4, 2011, 2:01 pm

Good tutorial thanks for sharing dude!

BauussHacker - May 4, 2011, 2:06 pm

(May 4, 2011 02:01 PM)Logic_Bomber Wrote: [\[CITE\]](#) Good tutorial thanks for sharing dude!

Thank you a lot. Finally someone replied, is there anything I should add, edit or remove? ?

Logic_Bomber - May 5, 2011, 11:24 am

I'm not sure but i will show it to a couple of friends and ask them if they think u should add anything

tflo - May 5, 2011, 3:51 pm

Lol look at this:

<http://www.elitepvpers.de/forum/co2-programming/1163559-tut-make-your-own-simple-c-compiler.html>.

Redeemz - May 5, 2011, 3:58 pm

Great work as always! Keep it up this was a good, useful idea. I will not be making one, but there are things in here i can reference later on for future projects :)

Zer0Fact0r - May 5, 2011, 4:10 pm

What you can improve? Your code in general. It's quite horrid.

Also in case anyone gets the wrong idea, you're not actually making a compiler. You're just making your own editor that uses Microsoft's C# compiler.

Odinbiorn - July 23, 2011, 2:56 am

Very interesting.

Perhaps you are not getting many replies per views due to the title being miss-leading. I apologise if I am wrong, but would this not be creating your own IDE, rather than compiler?

I notice you use Codedom in this, would it be possible for you to create a tutorial on using codedom? (very interesting tool that I can't find much documentation for which is not for vb.net)

Ethereal - July 23, 2011, 3:09 am

(Jul 23, 2011 02:56 AM)Odinbiorn Wrote: [\[CITE\]](#) Very interesting.

Perhaps you are not getting many replies per views due to the title being miss-leading. I apologise if I am wrong, but would this not be creating your own IDE, rather than compiler?

I notice you use Codedom in this, would it be possible for you to create a tutorial on using codedom? (very interesting tool that I can't find much documentation for which is not for vb.net)

Yes, you are right, but you are also gravedigging.