

Re: how does this still work??

Source:

<http://www.tech-archive.net/Archive/DotNet/microsoft.public.dotnet.languages.csharp/2005-10/msg01417.html>

- *From:* "Kevin Spencer" <kevin@xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx>
 - *Date:* Sun, 16 Oct 2005 07:31:52 -0400
-

Hi Barry,

Your concern is commendable. You will never become a good programmer if you can't understand code that is written for you by a designer. Visual Studio is an awesome tool set, and contains a great variety of devices to help you write code. The danger is that people who use it will let it become a crutch for them, and when the time comes that they have to do something it does NOT do for them, they will be dead in the water.

Visual Studio includes a set of Templates, which are basically files of pre-written code. When you create a new project of a certain type, the template for that project, which contains code that is common to that type of project, is copied for your project. This saves you the trouble of having to type in the basic using statements, class definitions, etc. It gives you a ready-made starting point.

Visual Studio also includes a large set of visual Designers. The Designer, for example, for a Windows Form, reads the code, and displays a graphic representation of the graphic user interface that the code will create at run-time. The Properties window is simply a set of the properties of the selected class instance in the Designer, and it uses Reflection to determine what properties are available to pre-set for that instance. When you type in a property in the Properties window, VS.Net writes the code for you in the class definition. This is also a real time-saver, as it saves you the time and trouble of writing out the property definition yourself, and prevents typing errors to a great extent, one of the most common causes of bugs in the software.

Add to that Intellisense, auto-indenting, and a host of other productivity tools, and you have a great toolkit that automates a lot of what would be hand-typing, and looking up class references in the SDK all day long. So, it's a good thing to have.

However, again, a tool is not a substitute for knowledge and understanding. And, in fact, you can use this toolset to learn how to program with C# as a bonus. Here's how:

When the Designer writes code for you, look at the code it writes. Study how

Re: how does this still work??

it is written. Why, for example, did the Template include the following?

```
Static void Main ( )  
{  
Application.Run(new MyForm( )); // nor this  
}
```

Every program has an entry point, a point where execution begins. In a C# program, this is the method called Main. It is the entry point because of its name. The compiler looks for a method called Main, and sets it as the entry point for the application.

Or, how about this?

```
//txtEnter  
this.txtEnter.Location = new System.Drawing.Point(16,32);  
this.txtEnter.Size = new System.Drawing.Size(264,20);
```

Well, in the designer, you placed a textbox on the form. You may have sized it, or left the default Designer size, but it did have a size. A visual Control in a form is drawn at a location on the form. The location is determined by an XY coordinate relative to the upper left-hand corner of the Control it is immediately inside (in this case, the form itself). This is how the OS knows how and where to draw the Control. The Point structure was created specifically for identifying an XY coordinate, or location, in 2-dimensional space. Similarly, the Size structure was created to identify the dimensions (width and height) of a rectangle in 2-dimensional space. In the case of a textbox, the inside is simply scaled to fit inside that space at that point.

By studying the code that the Designer writes for you, you can learn how these things are programmed, and learn to program them for yourself in the future, if need be. As an experiment, try changing the dimensions in the code, and switch to the Designer view to see what the result is. It should be fun.

Bottom line, your first experiment was a success! Keep going, and try to take on a little at a time. It should be fun. The science of programming is an elephant, but even an elephant can be eaten, one byte at a time!

—

HTH,

Kevin Spencer
Microsoft MVP
..Net Developer
Ambiguity has a certain quality to it.

"tindog" <tindog@xxxxxxxxxxx> wrote in message
news:4351f523@xxxxxxxxxxxxxxxxxxxxxxx

Re: how does this still work??

Re: how does this still work??

>I seem to be caught in a bit of a conundrum with C#. First of all getting
>books on VS 2003.net then VS 2005 comes out and further a book I have
>bought to just learn just the language C# (in 21 days). I now have ordered,
>as suggested to me by a few members of the newsgroup to get a Primer type
>book how ever until then I will use what I have. BUT looking at the C#
>language my first exercise which happens to be a windows form looks quite
>scary. I got this to work "but why do I put in a tiny bit of the code in
>and it works. I'll explain what I did. What I used was a win app form in
>VS2005. The object is to write in the text box press enter button and what
>have printed goes to the label. Rewrite text in text box and press enter
>and what entered also goes into label box but on top of previous message.

>
> This is source code with my comments, I understand this is written so can
> be compiled in a basic compiler via command line.

>

>

>

> Hello through windows form:

>

> Using System;

>

> Using System.Windows.Forms; // don't have to do anything her already
> written plus a /// few more systems etc.. on VS 2005

>

>

>

> namespace HelloWin

>

> {

>

> public class MyForm : Form // vs2005 public partial class Form1 : Form

>

> {

>

> private TextBox txtEnter; // These can be altered in properties box no
> probs here

>

> private Label lblDisplay;

>

> private Button btnOk;

>

>

>

> public MyForm() // public Form1() .suppose this can be altered in code
> anyway no //probs here

>

> {

>

> this.txtEnter = new TextBox(); // did not enter this in vs2005

>

> this.lblDisplay = new Label(); // nor this

Re: how does this still work??

Re: how does this still work??

```
>
> this.btnOk = new Button (); // nor this
>
> this.Text = "My HelloWin App!"; // entered in properties box for form and
> became title //at top of form
>
> //txtEnter
>
> this.txtEnter.Location = new System.Drawing.Point(16,32); // entered in
> prop box for text //box
>
> this.txtEnter.Size = new System.Drawing.Size(264,20); // entered in prop
> box
>
>
>
> // lblDisplay
>
> this.lblDisplay.Location = new System.Drawing.Point(16,72); // prop box
> for label
>
> this.lblDisplay.Size = new System.Drawing.Size(264,128); // can control
> from vs2003 //but not vs2005 always defaults to 31,31 but somehow
> stretches as wanted to.
>
>
>
> // btnOk
>
> this.btnOk.Location = new System.Drawing.Point(88,224); // control from
> prop box
>
> this.btnOk.Text = "OK"; // control from property box
>
> this.btnOk.Click += new System.EventHandler(this.btnOk_Click); // did not
> have to enter this
>
>
>
> // myform
>
> this.Controls.AddRange(new Control [ ] {
>
> this.txtEnter, this.lblDisplay, this.btnOk}); // nor this
>
> }
>
> Static void Main ( ) //nor this
>
> {
>
```

Re: how does this still work??

Re: how does this still work??

```
> Application.Run(new MyForm( )); // nor this
>
> }
>
>
>
> Private void btnOk_Click(object sender, System.EventArgs e) //
> automatically inserted
>
> {
>
> lblDisplay.Text = txtEnter.Text + "\n" + lblDisplay.Text; // only code I
> had to enter and it worked (Why)???
>
> }
>
> }
>
> }
>
>
> Whew glad that's over.. What I did was that okay ??? or was there syntax I
> should have put in??? (even though it did work)..
>
>
>
> Barry
>
>
```

• **References:**

◆ **[how does this still work??](#)**

◇ *From:* tindog

• Prev by Date: **[Re: cross-platform programs](#)**

• Next by Date: **[Re: Sniffing Network Data](#)**

• Previous by thread: **[Re: how does this still work??](#)**

• Next by thread: **[Re: how does this still work??](#)**

• Index(es):

◆ **[Date](#)**

◆ **[Thread](#)**