

## Visual C# Program: Resistor Sizing Calculator

---

**In this chapter, you will learn how to use the following Visual C# Application functions to World Class standards:**

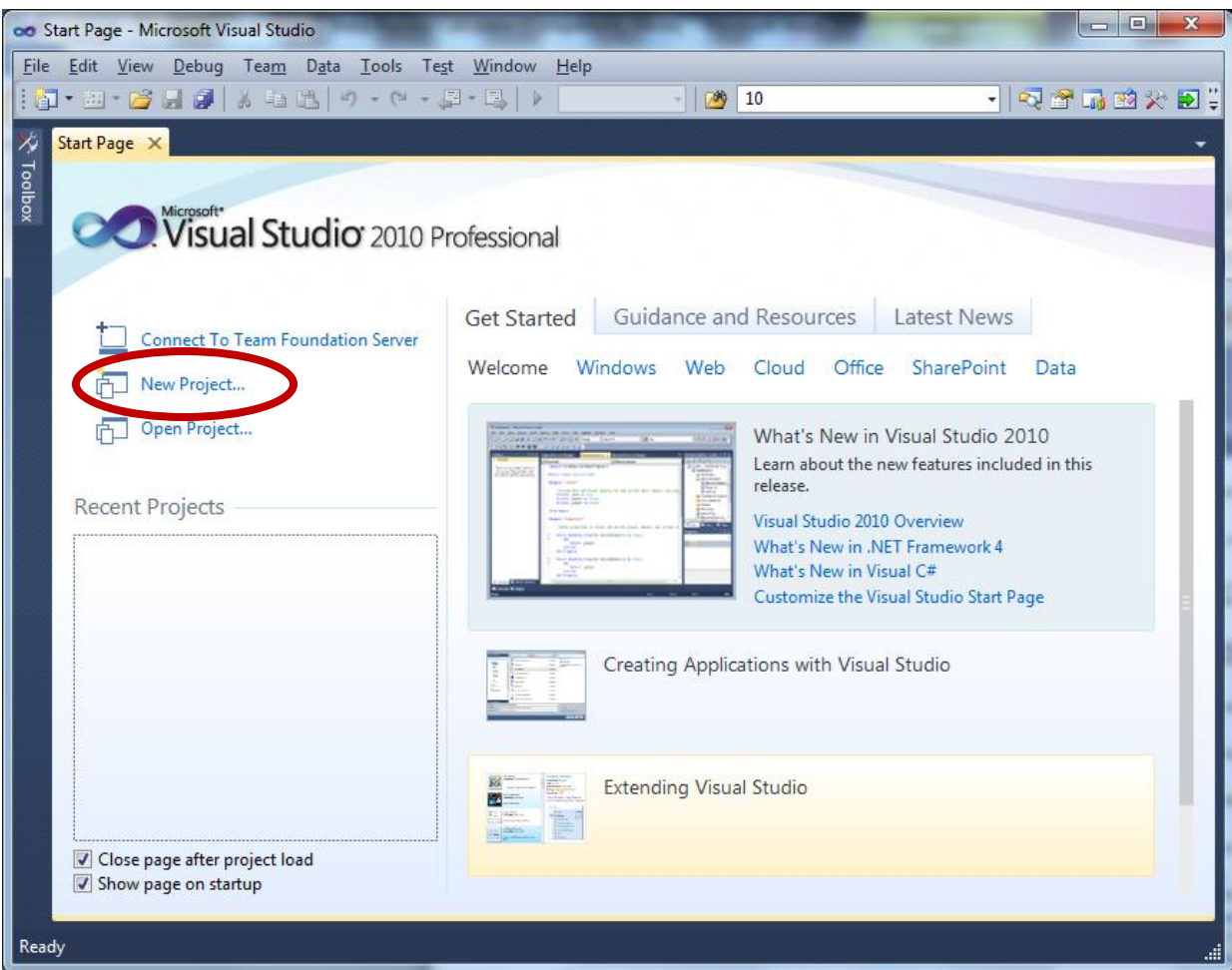
- **Opening Visual C# Editor**
- **Beginning a New Visual C# Project**
- **Laying Out a User Input Form in Visual C#**
- **Insert a Label into a Form**
- **Insert a Textbox into a Form**
- **Insert a Label into a Form to Post an Output**
- **Insert Command Buttons into a Form**
- **Adding a Copyright Statement to a Form**
- **Insert a Picture into a Form**
- **Adding Comments in Visual C# to Communicate the Copyright**
- **Declaring Variables in a Program**
- **Setting Variables in a Program**
- **Using a Label to Communicate with Variables**
- **Ending the Program**
- **Running the Program**

## Open the Visual C# Editor

---

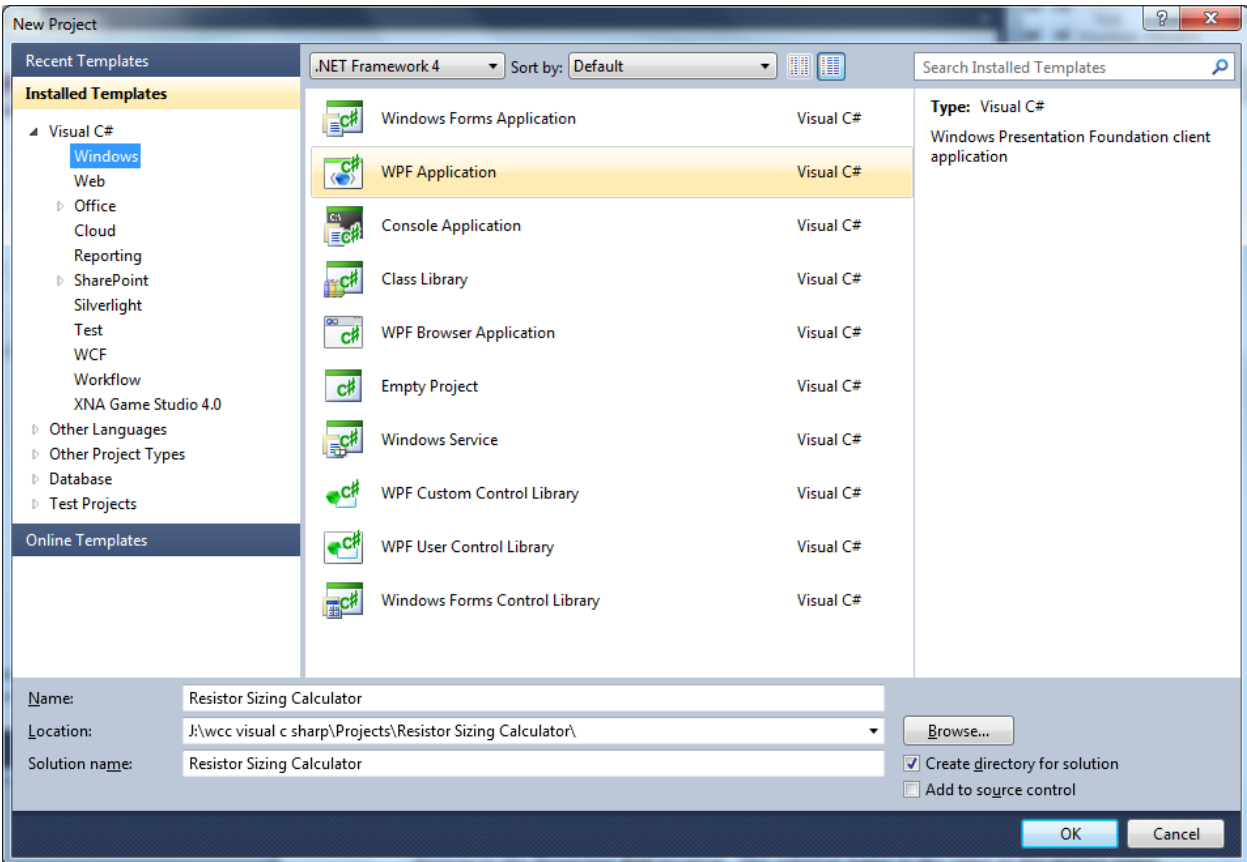
In our next lesson, we will step through each procedure in adding labels, textboxes and command buttons and we will integrate into the tutorial the methods to add, subtract, multiply and divide numbers. We will also include formatting the answers as they are shown in the answer labels. As in every project, we will create variable, set their values, execute mathematical equations and output data. In this lesson, we revisit the procedure to add the computer date and time to the form.

To open a new project, we select New Project on the left side of the Visual Studio window.



**Figure 4.1 – The Start Page**

We start a new Windows Application by picking the Windows Application icon from the installed templates list on the New Project window.



**Figure 4.2 – New Project**

We start a new Windows Application Project by picking the Windows under Visual C # in the left pan of the New Project window. Then we pick Windows Form Application in the center pane.

At the bottom of the Window, we name the project, Resistor Sizing Calculator. We make a folder for our projects called Visual C Sharp on the desktop, on our flash drive or in the Documents folder. We make another folder inside the first called Resistor Sizing Calculator. On the New Project window, we browse to the Resistor Sizing Calculator location. The solution name is the same as the project name.

## Beginning a New Visual C# Application

Remember, that all programming projects begin with one or more sketches. The sketch will show labels, textboxes, and command buttons. In this project, we will name the input form, **Resistor Sizing Calculator**. We will have two textboxes to key in the voltage and current that the resistor will receive and both textboxes will have labels to identify their content. We will have two labels with borders to output the resistance and power ratings of the resistor. Those two boxes will have labels for identification. We will have three command buttons, **Calculate**, **Reset** and **Exit**. On the bottom of the form, we will write the copyright statement using another

label. On this presentation, we can help ourselves by being as accurate as possible, by displaying sizes, fonts, colors and any other specific details which will enable us to quickly create the form. On this form, we will use a 12 point Arial font. We will have a graphic of a resistance and power formulas above the Exit button. From the beginning of inserting the form into the project, we need to refer to our sketch.

We should train new programmers initially in the art of form building. When using the editor, we insert and size the form, and selecting the Controls Toolbox, we will place all the various input tools and properly label them. Whenever we place an input tool, the properties window will display a list of every attribute associated with the tool, and we will take every effort to arrange the tool by performing such actions as naming, labeling and sizing the visual input device.

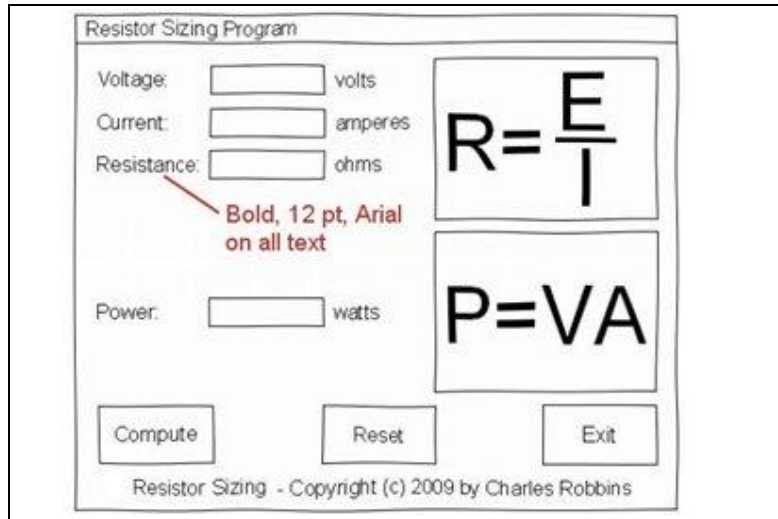


Figure 4.3 – Sketch of the Resistor Sizing Form

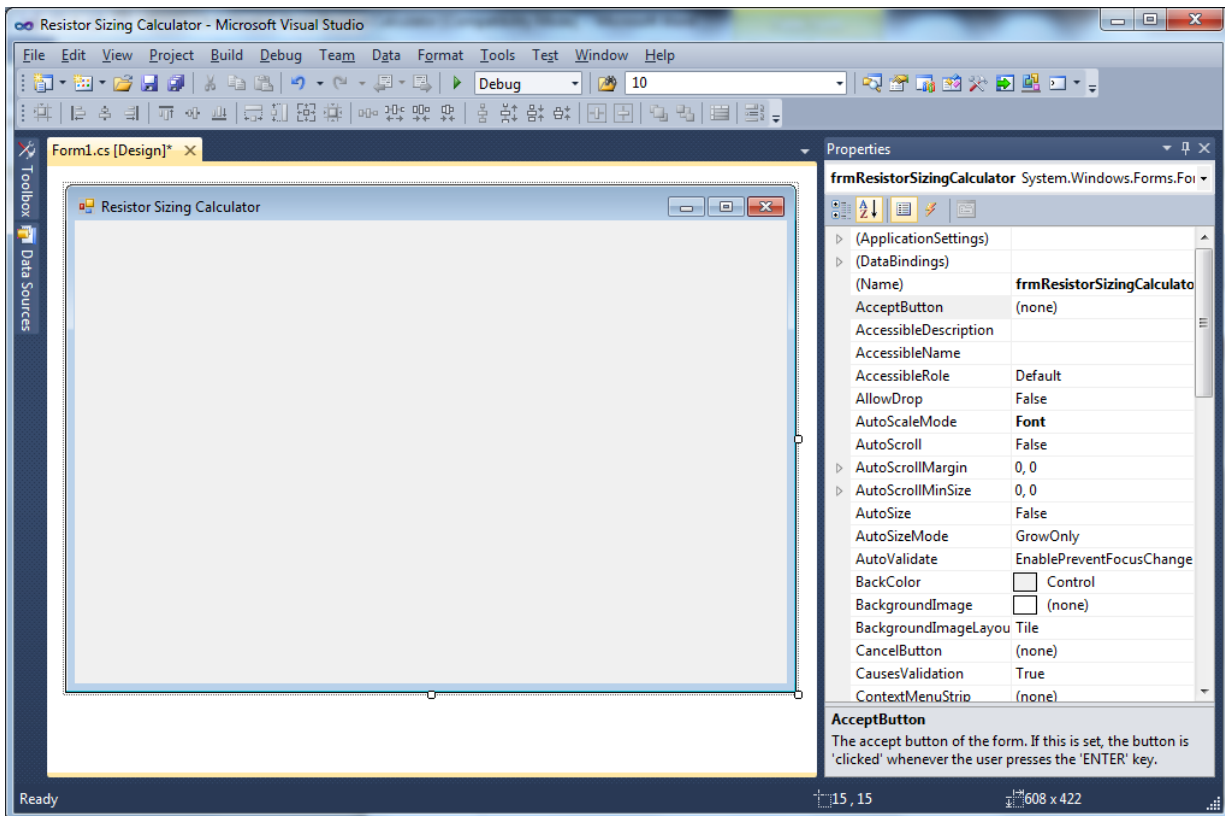


Figure 4.4 – Designing the Resistor Sizing Calculator Form in Visual C#

## Laying Out a User Input Form in Visual C#

We will change the **Text** in the Properties pane to Resistor Sizing Calculator to agree with the sketch in Figure 4.3. Go ahead and change the form in two other aspects, BackColor and Size.

Alphabetic	
BackColor	White
Font	Arial, 12 pt
Size	477,313

The first number is the width and the second number is the height. The form will change in shape to the size measurement.

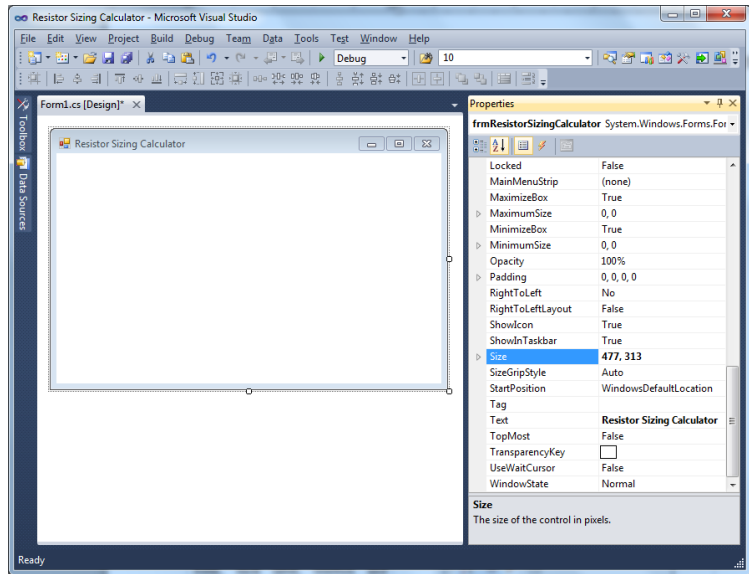


Figure 4.5 – Setting the Form Properties

The background color will change to a white. There are many more attributes in the Properties pane that we will use on future projects.

In this project, we will select the font in the form. By selecting the font, font style and size for the form, each label, textbox and command button we insert will have these settings for their font.

When highlighting the row for Font, a small command button with three small dots appears to the right of the default font name of Microsoft San Serif. Click on the three dotted button to open the Visual C# Font window.

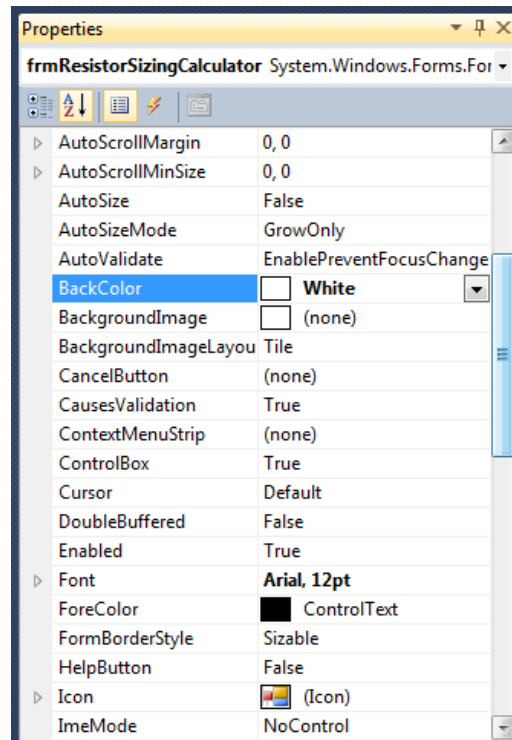


Figure 4.6 – The Font Window in Visual C#

We will select the Arial font, Regular font style and 12 size for this project to agree with the initial sketch if the user input form. If we wish to underline the text or phrase in the label, add a check to the Underline checkbox in the Effects section of the Font window. When we finish making changes to the font property, select the OK command button to return to the work area.

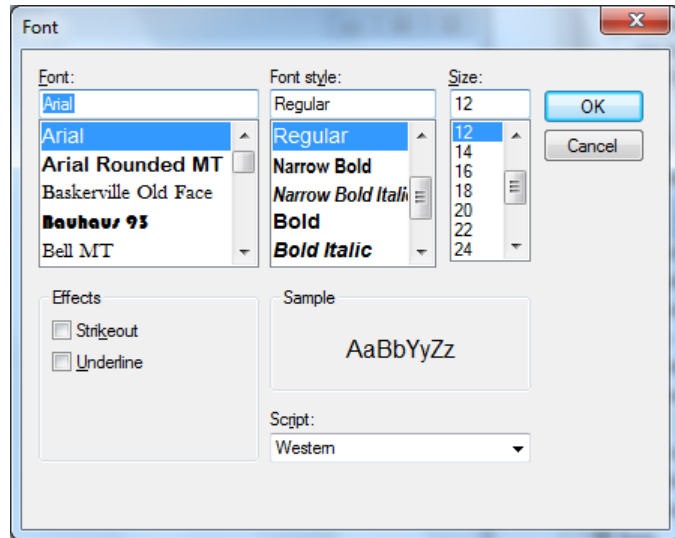


Figure 4.7 – Changing the Font to Arial

## Inserting a Label into a Form

A good form is easy to figure out by the user, so when we are attempting to provide information on the window that will run in Windows; we add labels to textboxes to explain our intent. Press the Label (A) button on the Control Toolbar to add a label. To size the label area, click on the upper left area of the form and hold down on the left mouse button, draw the dotted label box.

When the first label is done, the background color of the label matches the background color of the form. In many cases that effect is visually pleasing to the eye, versus introducing another color. Both color and shape will direct the user in completing the form along with the explanation we place on the window to guide the designer in using the automated programs. Use colors and shape strategically to communicate well.

We will insert our first Label on the upper left corner of the form and call the entity **lblVoltage**.

Alphabetic	
(Name)	lblVoltage
BackColor	White
Text	Voltage
Font	Arial, 12 pt

Since the backcolor and font are already set, we just type “Voltage” at the text attribute.

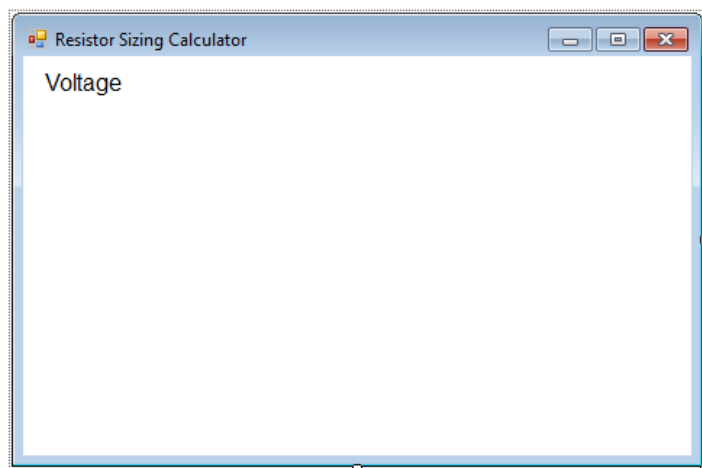


Figure 4.8 – The Finished Label on the Form

## Inserting a Textbox into a Form

A textbox is used so that a user of the computer program can input data in the form of words, numbers or a mixture of both. Press the TextBox (ab) button on the Control Toolbar to add a textbox. To size the label area, click on the upper left area of the form and hold down on the left mouse button, draw the dotted textbox.

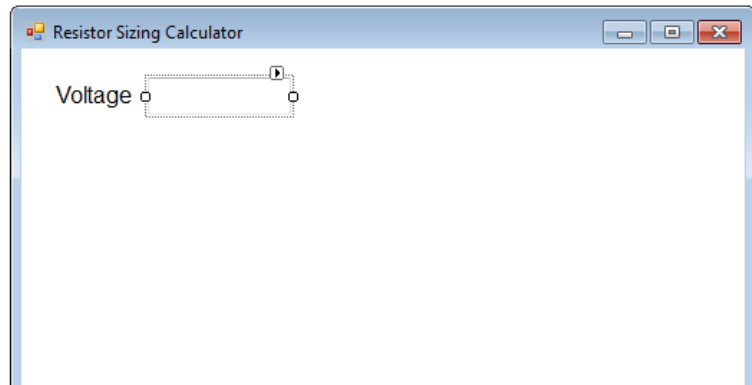


Figure 4.9 – Placing a TextBox on the Form

We will name the TextBox using the three letter prefix followed by the name or phrase of the tool. For our first textbox, the name is **txtVoltage**.

Alphabetic	
(Name)	txtVoltage
Size	100, 26

The size of the textbox will be 100 wide and 26 tall and the characters inside the textbox will be aligned to the right.

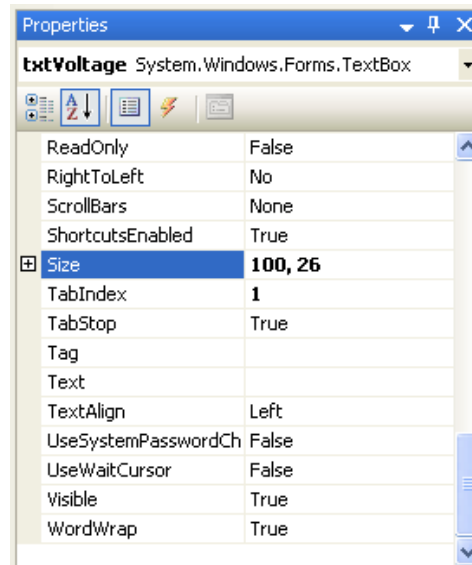


Figure 4.10 – Setting the Size of the TextBox

We will place a label to the right of the **txtVoltage** textbox and call it **lblVolts**. We will make the label text Volts. The key attributes for the label are:

Alphabetic	
(Name)	lblVolts
BackColor	White
Text	Volts
Font	Arial, 12 pt

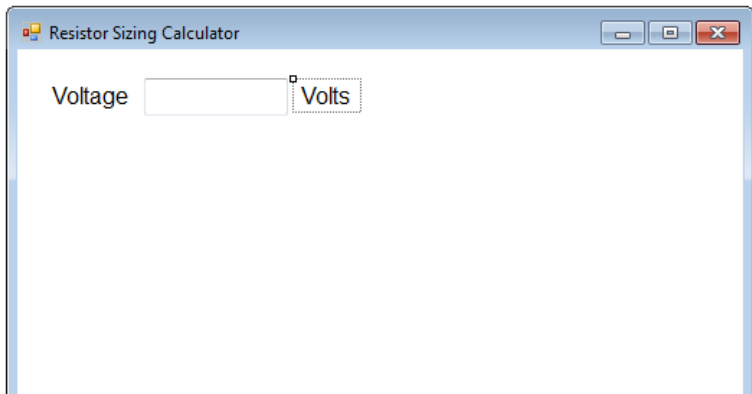


Figure 4.11 – Adding another Label

We add another row of label, textbox and label for the current. The following are the key properties.

Alphabetic	
(Name)	lblCurrent
BackColor	White
Text	Current
Font	Arial, 12 pt

Alphabetic	
(Name)	txtCurrent
Size	100, 26

Alphabetic	
(Name)	lblAmperes
BackColor	White
Text	amperes
Font	Arial, 12 pt

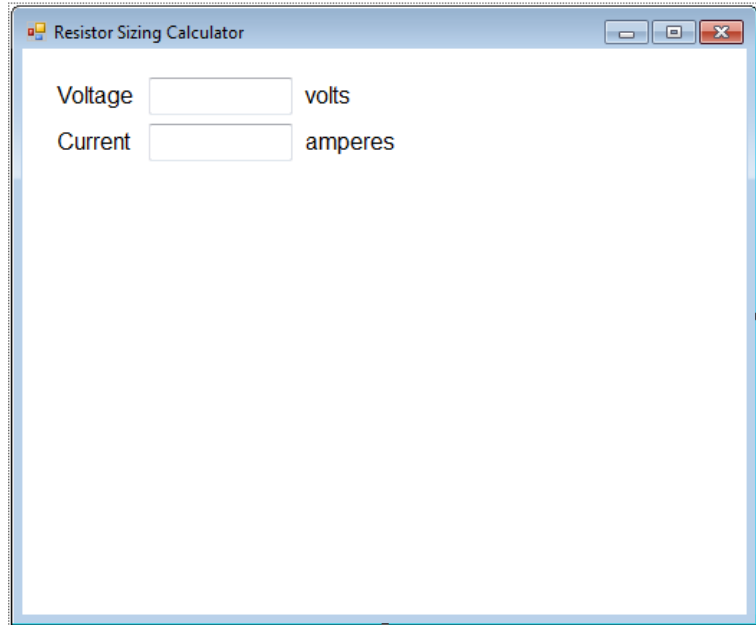


Figure 4.12 – Second Row of Labels and Textbox

## Inserting a Label into a Form to Post the Output

Some labels on a form are in a position to display an answer after the user inputs data and they press the command button to execute the application. To add this label, press the Label (A) button on the Control Toolbar to add a label. To size the label area, click on the upper left area of the form and hold down on the left mouse button, draw the dotted label box.

We will place a label under **lblCurrent** label and call it **lblResistance**. We will make the label text Resistance. The key attributes for the label are:

Alphabetic	
(Name)	lblResistance
BackColor	White
Text	Resistance
Font	Arial, 12 pt

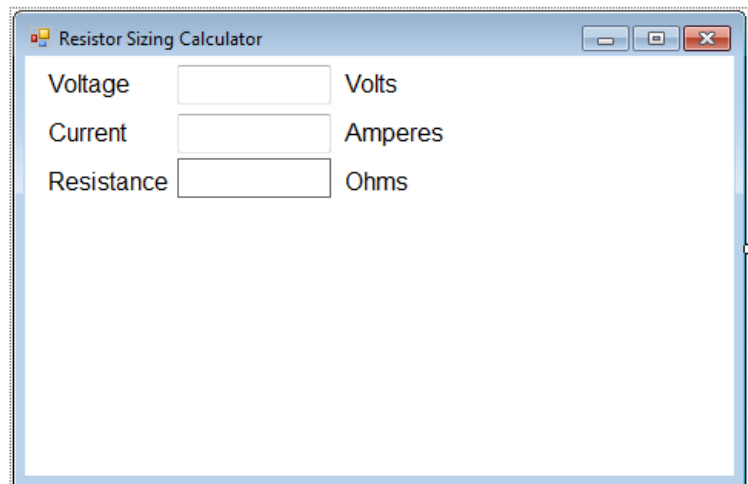


Figure 4.13 – Placing a Label for the Answer



We will insert the label for the answer to the right of **lblResistance** and name the label **lblOhmsAnswer**.

Alphabetic	
(Name)	lblOhmsAnswer
AutoSize	False
BorderStyle	FixedSingle
Size	100,26
TextAlign	Middle left

We will make **AutoSize** False to allow the label to size to 100 by 26 and the **borderstyle** **FixedSingle** to place a line around the answer.

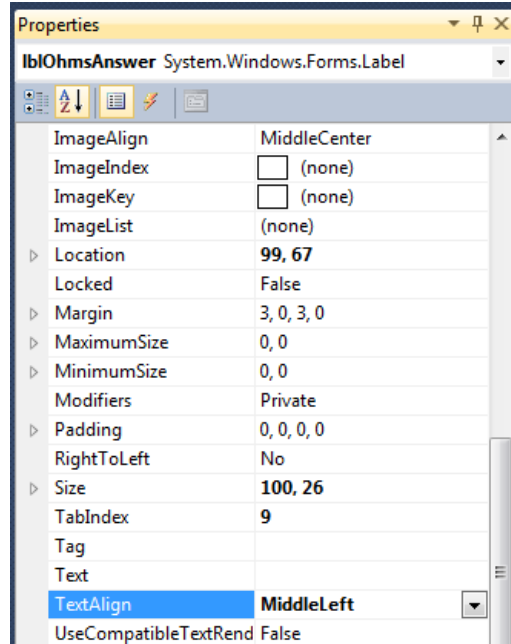


Figure 4.14 – Label Name is lblOhmsAnswer

We will place a label to the right of the **lblOhmsAnswer** textbox and call it **lblOhms**. We will make the label text **Volts**. The key attributes for the label are:

Alphabetic	
(Name)	lblOhms
BackColor	White
Text	Ohms
Font	Arial, 12 pt

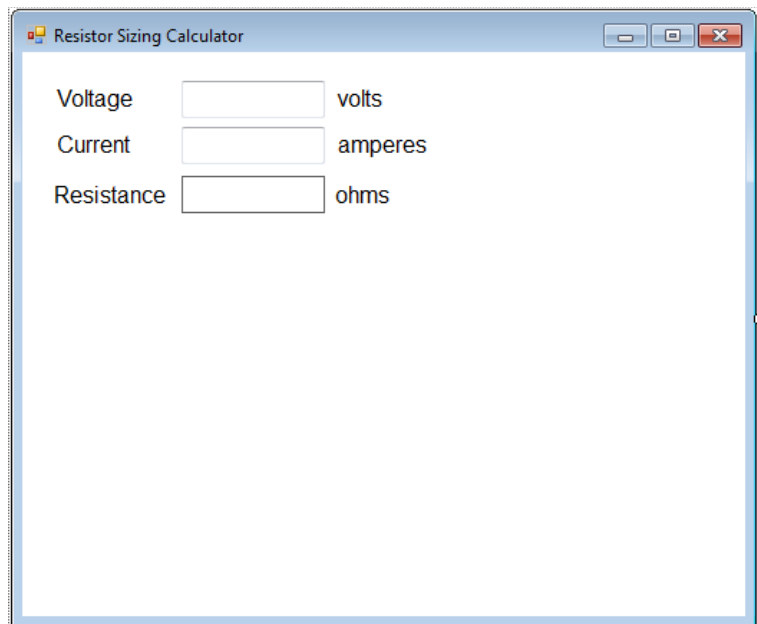


Figure 4.15 – Label Name is lblOhms

We will place a label that contains an answer for the resistor power in a fourth row. Again, it will start with an identifying label, then a label with a border and lastly a label that describes the unit of measurement.

We will place a label under **lblResistance** label and call it **lblPower**. We will make the label text Power. The key attributes for the label are:

Alphabetic	
(Name)	lblPower
BackColor	White
Text	Power
Font	Arial, 12 pt

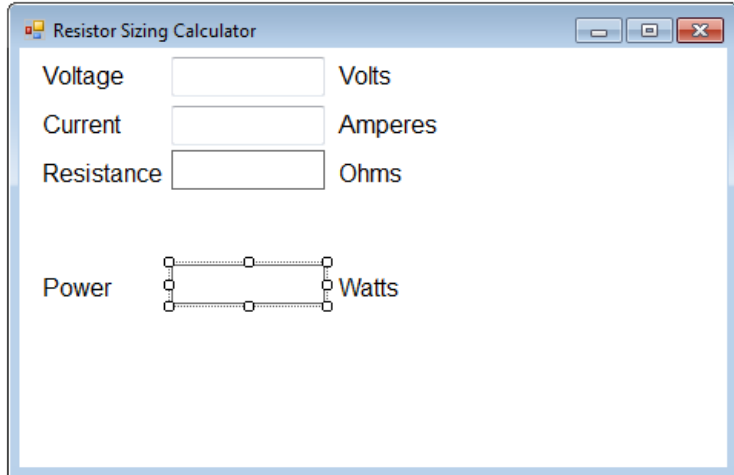


Figure 4.16 – Label Name is lblPower

We will insert the label for the answer to the right of **lblPower** and name the label **lblWattsAnswer**.

Alphabetic	
(Name)	lblWattsAnswer
AutoSize	False
BorderStyle	FixedSingle
Size	100,26
TextAlign	Middle left

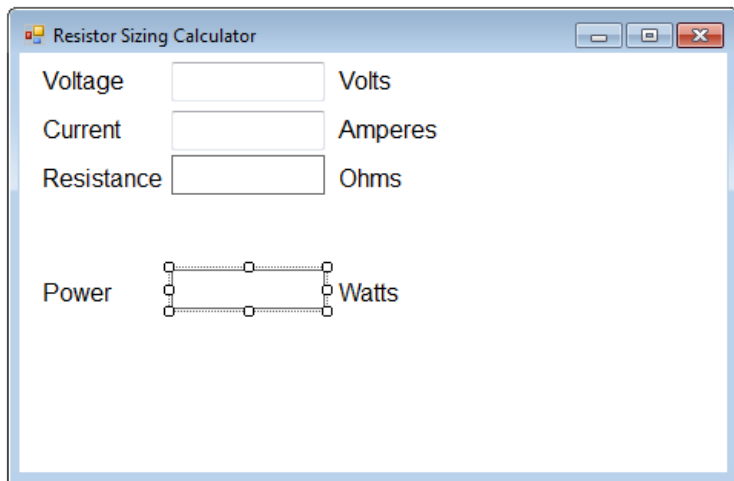


Figure 4.17 – Label Name is lblWattsAnswer

We will place a label to the right of the **lblWattsAnswer** textbox and call it **lblWatts**. We will make the label text Watts. The key attributes for the label are:

Alphabetic	
(Name)	lblWatts
BackColor	White
Text	Watts
Font	Arial, 12 pt

## Inserting a Command Buttons into a Form

---

A command button is used so that a user will execute the application. Press the Command button on the Control Toolbar to add a command button. To size the label area, click on the upper left area of the form and hold down on the left mouse button, draw the command button as shown in Figure 4.18.

We will name the command button using the name is **cmdCalculate**.

Alphabetic	
(Name)	cmdCalculate
Caption	Calculate
Font	Arial, 15.75 pt
Size	133,33

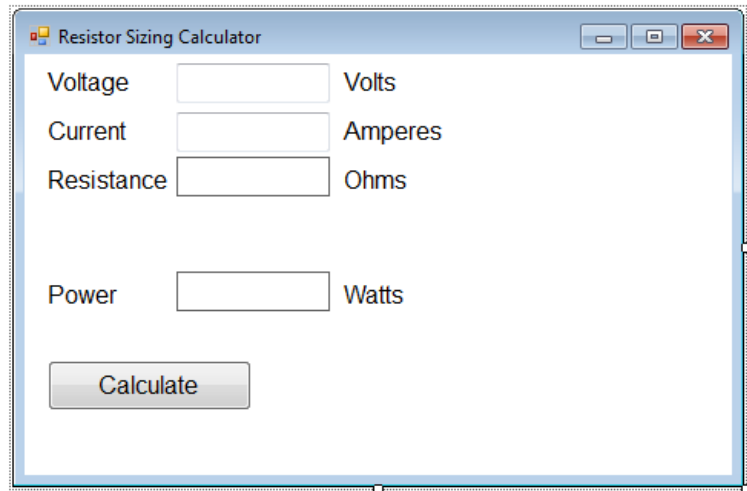


Figure 4.18 – The Command cmdCalculate Button

Add a second Command button, named cmdReset is for clearing the txtName and lblGreeting objects. The third command button is to exit the program. When the user presses the Exit command button, the application closes. Notice the equal spacing between the command buttons gives a visually friendly appearance.

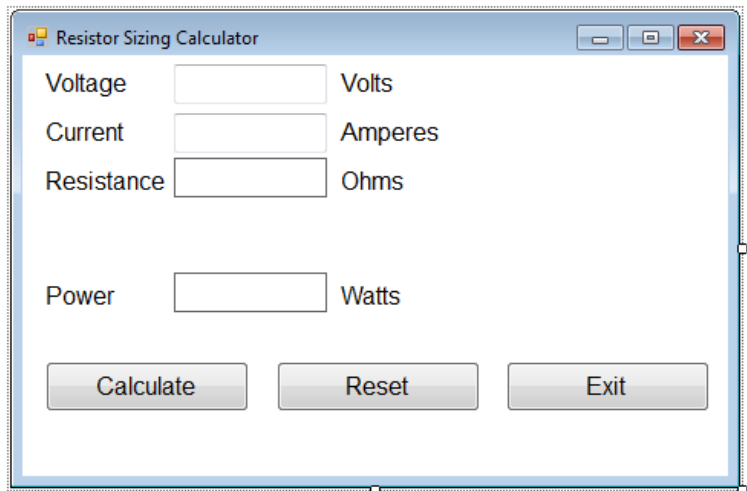


Figure 4.19 – Insert Two More Command Buttons

## Adding a Copyright Statement to a Form

---

At the beginning of a new program, we will expect to see an explanation or any special instructions in the form of comments such as copyright, permissions or other legal notices to

inform programmers what are the rules dealing with running the code. Comments at the opening of the code could help an individual determine whether the program is right for their application or is legal to use. The message box is a great tool when properly utilized to inform someone if they are breaking a copyright law when running the code.

Finish the form with the following copyright information.

### Resistor Sizing Calculator - Copyright 2011 by charles robbins

If there are special rules or instructions that the user needs to know, place that information on the bottom of the form.

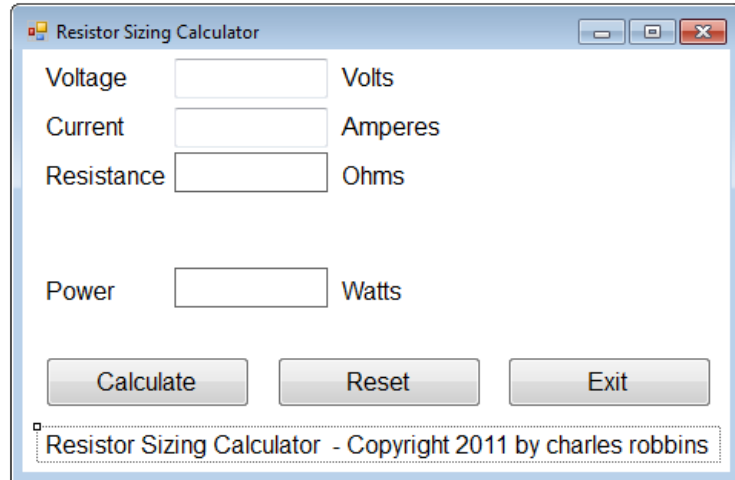


Figure 4.20 – Adding a Copyright Statement

## Inserting a Picture into a Form

We select the toolbox and PictureBox and we draw a box to the right of the labels that will contain the answers. We name the picturebox **imgFormula**. We scroll down on the properties window and select the three dots button at the Image property and a Select Resource window will appear.

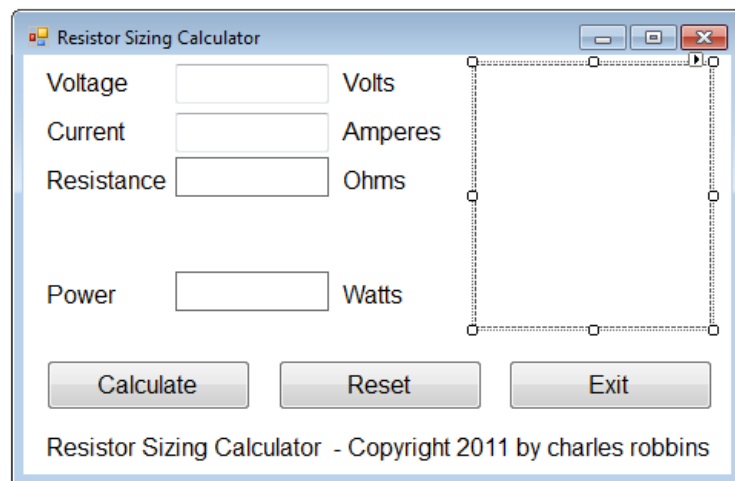


Figure 4.21 – Adding an Image

We then will import the graphic of our gas pump which we made in Microsoft Paint and saved as a bitmap image. We then press the OK button and the image will appear in the picturebox.

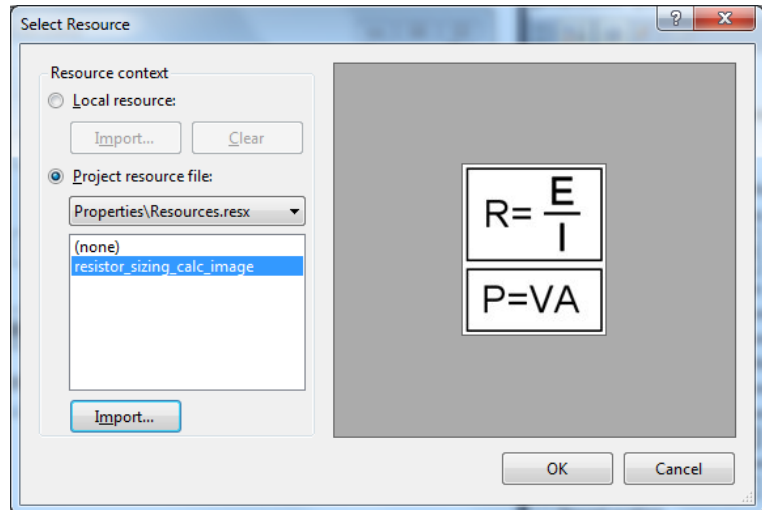


Figure 4.22 – Import an Image

The finished form is as shown in figure 4.23. Now, we will start to write the code to find the resistance and power of the resistor.

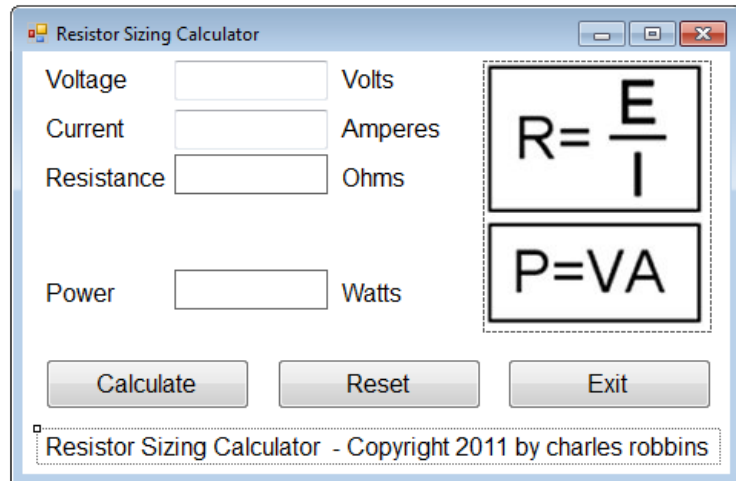


Figure 4.23 – Import an Image

## Adding Comments in Visual C# to Communicate the Copyright

The comments we placed in the first three lines of the program will inform the individual opening and reading the code of the ownership. This is for those user that may run the application without checking the label on the bottom of the form with the copyright information. It is a great tool to alert the client to the rules of the program and tell them what the application will do.

To begin the actual coding of the program, double click on the form. At the top of the program and after the line of code with `Namespace Resistor_Sizing_Calculator {`, place the following comments with two slash (`//`) characters. Remember, the two slashes (`//`) will precede a comment and when the code is compiled, comments are ignored.

Type the following line of code:

```
//Resistor Sizing Calculator - Copyright (c) 2011 by Charles W. Robbins  
//This program will open a dialogue box, allow the user to type voltage and current for a resistor  
//When the user clicks on the Calculate button, the resistance and power is calculated.
```

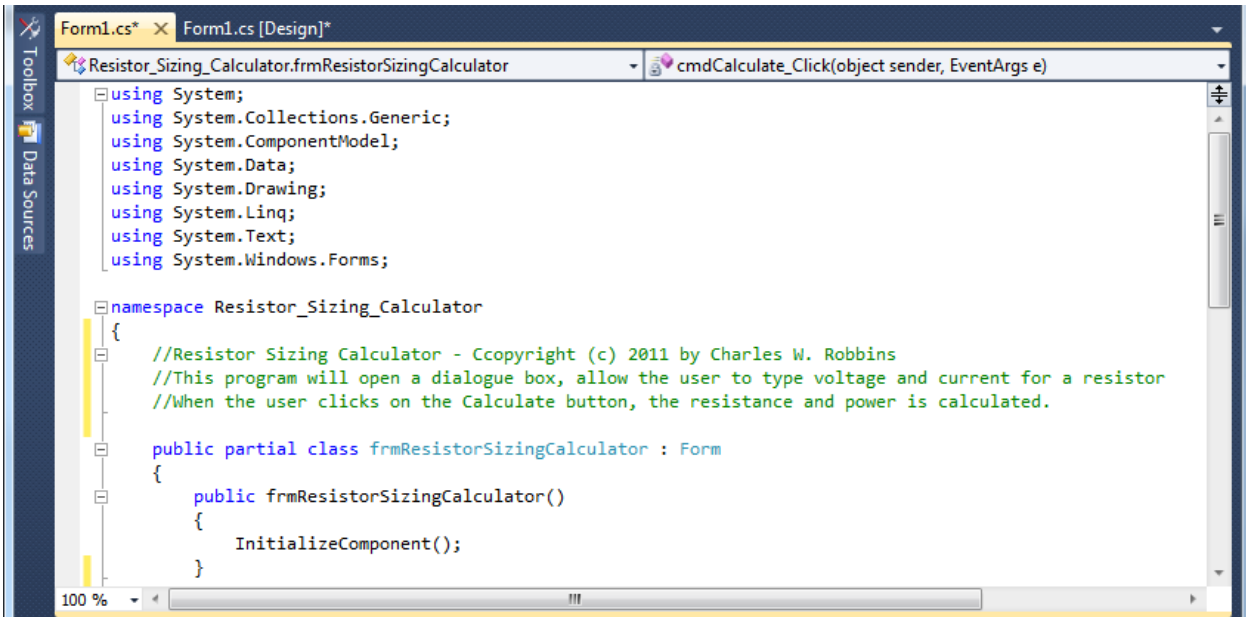


Figure 4.24 – Adding a Copyright Statement

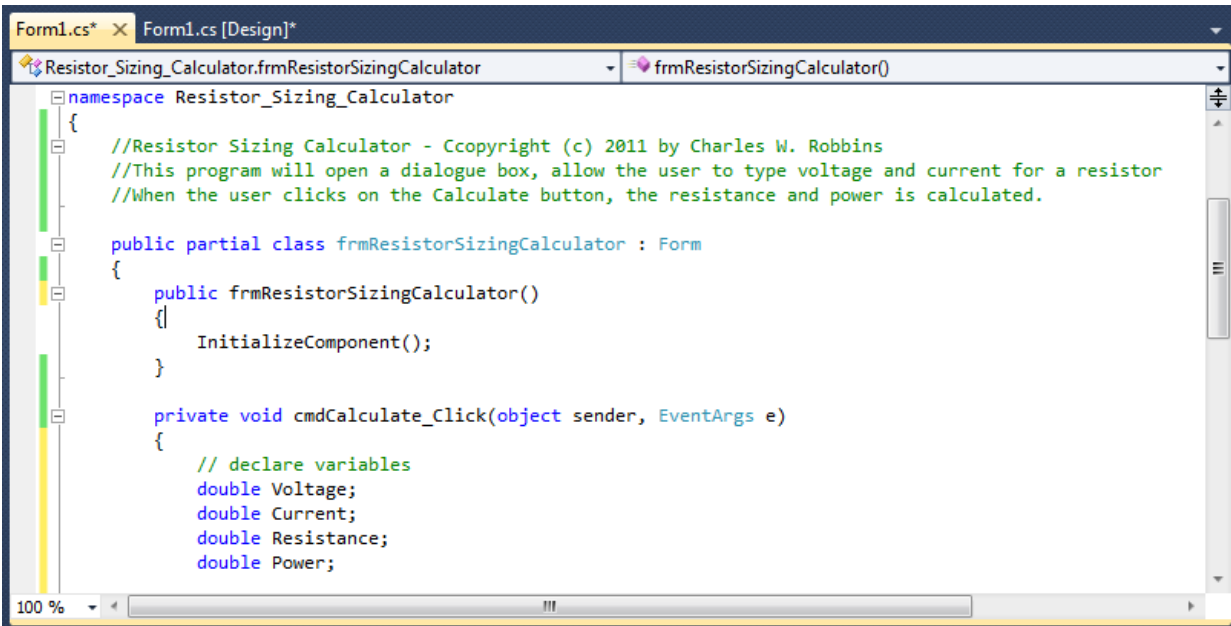
## Declaring Variables in a Program

When we are going to use a number, text string or object that may change throughout the life of the code, we create a variable to hold the value of that changing entity. In Visual C#, the integer statement is one of the ways to declare a variable at the procedure level.

In our program, we will retrieve the data from the textboxes and also we will create data from mathematical computations. We will place the values in variables called Voltage, Current, Resistance, and Watts. These variables will hold numbers for calculations so we will declare them as Integers.

Type the following code under the cmdCalculate subroutine of the program.

```
// declare variables  
double Voltage;  
double Current;  
double Resistance;  
double Power;
```



**Figure 4.25 – Declaring Variables with Dim Statements**

Notice that the variable name should be a word or a phrase without spaces that represents the value that the variable contains. If we want to hold a value of one's date of birth, we can call the variable, DateofBirth. The keywords Date and Birth are in sentence case with the first letter capitalized. There are no spaces in the name. Some programmers use the underscore character ( ) to separate words in phrases. This is acceptable, but a double underscore ( ) can cause errors if we do not detect the repeated character.

## Setting Variables in a Program

---

Next, we will set the variables using the equal function. We will set the numbers in the two textboxes to their variable and we compute resistance by division and the power by using multiplication.

Type the following code under the "set variable" section of the cmdCalculate subroutine of the program.

```
//Set variables  
Voltage = Convert.ToDouble(txtVoltage.Text);  
Current = Convert.ToDouble(txtCurrent.Text);  
Resistance = Voltage / Current;  
Power = Voltage * Current;
```

For the first our first try at setting variables, we type the expression

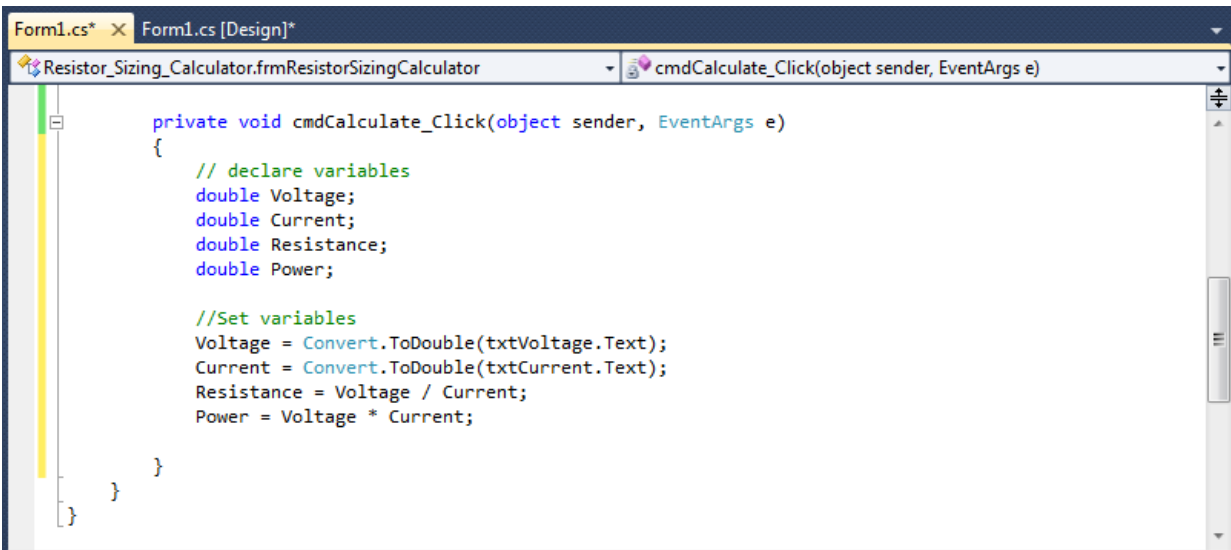
```
Voltage = Convert.ToDouble(txtVoltage.Text);
```

We use the Convert.ToDouble function to take the text string in the textbox txtVoltage and change it to a double integer. As a real number, we can perform math functions that could not be done on a text string. Remember, text and numbers in the textbox are just text strings.

In the next section of our code, we will use basic math functions. They are multiplying, and dividing.

Basic Math Function	C# Function
Multiplying	*
Dividing	/

We will use these in almost every program.



```
private void cmdCalculate_Click(object sender, EventArgs e)
{
    // declare variables
    double Voltage;
    double Current;
    double Resistance;
    double Power;

    //Set variables
    Voltage = Convert.ToDouble(txtVoltage.Text);
    Current = Convert.ToDouble(txtCurrent.Text);
    Resistance = Voltage / Current;
    Power = Voltage * Current;
}
}
```

Figure 4.26 – Setting the Variables in the C# Code

## Using a Label to Communicate with Variables

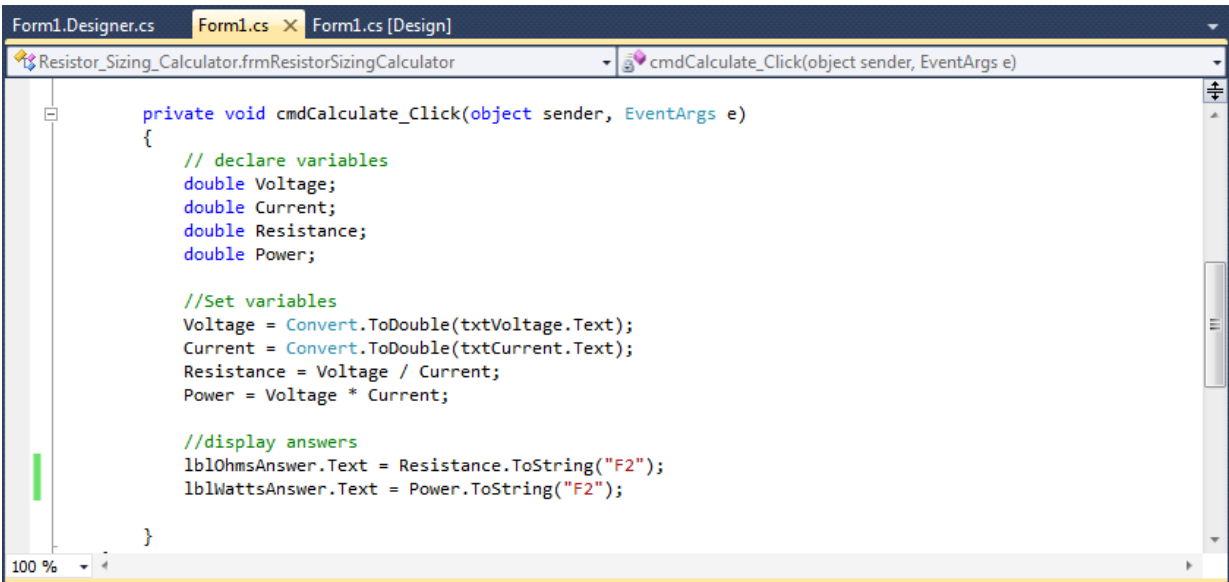
The numbers that were set to the variables can now be assigned to the answer labels using the equal sign such as **lblOhmsAnswer.Text = Resistance.ToString("F2")** and the answer will appear in the form with a single border around the numbers.

Go ahead and type the following code below the set variables section.

```
//display answers
lblOhmsAnswer.Text = Resistance.ToString("F2");
lblWattsAnswer.Text = Power.ToString("F2");
```

We have added a new feature to the output by using the ToString("F2") function that puts the number and then how many decimals we would like to see in the answer. In this case, we ask for two decimals.





```
private void cmdCalculate_Click(object sender, EventArgs e)
{
    // declare variables
    double Voltage;
    double Current;
    double Resistance;
    double Power;

    //Set variables
    Voltage = Convert.ToDouble(txtVoltage.Text);
    Current = Convert.ToDouble(txtCurrent.Text);
    Resistance = Voltage / Current;
    Power = Voltage * Current;

    //display answers
    lblOhmsAnswer.Text = Resistance.ToString("F2");
    lblWattsAnswer.Text = Power.ToString("F2");
}
```

Figure 4.27 – Displaying the Answers

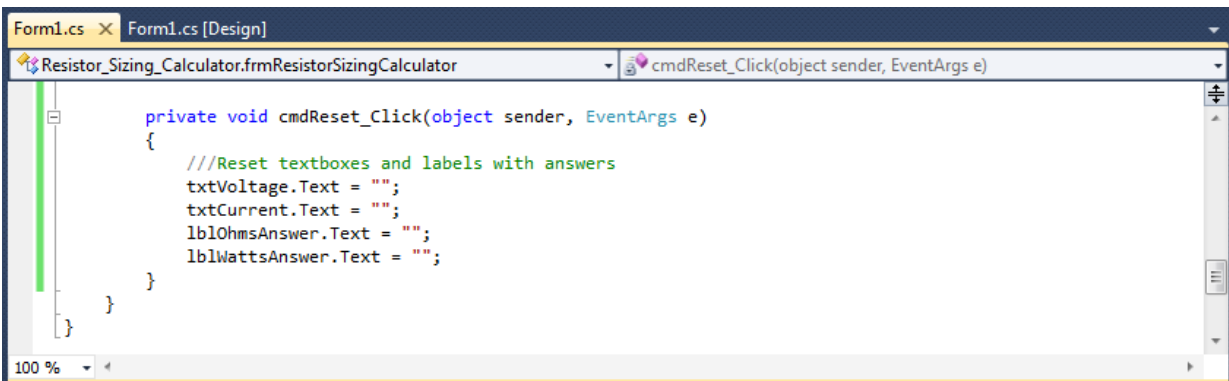
## Resetting the Data

---

To clear the textboxes or labels containing the data, we will replace the data with blank strings and the date and time with the current day and time setting.

Type the following code under the cmdReset subroutine of the program

```
//Reset textboxes and labels with answers
txtVoltage.Text = ""
txtCurrent.Text = ""
lblOhmsAnswer.Text = ""
lblWattsAnswer.Text = ""
```



```
private void cmdReset_Click(object sender, EventArgs e)
{
    ///Reset textboxes and labels with answers
    txtVoltage.Text = "";
    txtCurrent.Text = "";
    lblOhmsAnswer.Text = "";
    lblWattsAnswer.Text = "";
}
```

Figure 4.28 – Computing the Reset Button by Clearing a Textbox and Label Caption

## Exiting the Program

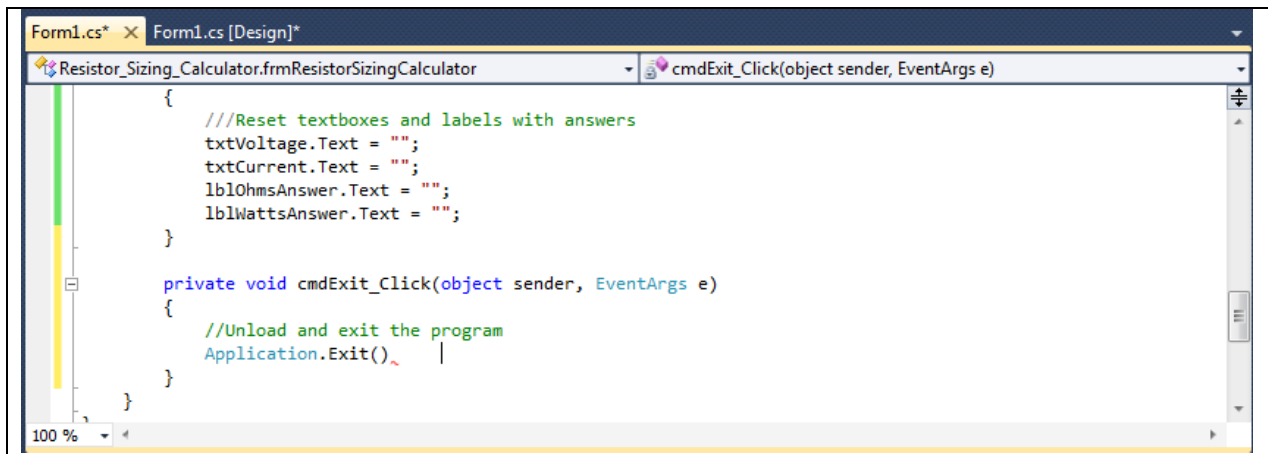


Figure 4.29 – Exiting the Program

To exit this program, we will unload the application and end the program.

Type the following code:

```
//Unload and exit the program
Application.Exit()
```

## Running the Program

After noting that the program is saved, press the F5 to run the Resistor Sizing Calculator application. The Resistor Sizing Calculator window will appear on the graphical display as shown in Figure 4.30. Notice the professional appearance and presentation of information in a clean dialogue box.

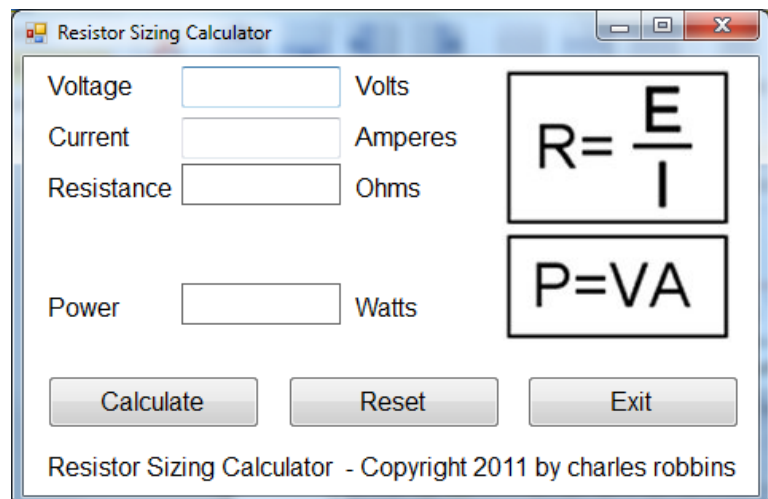
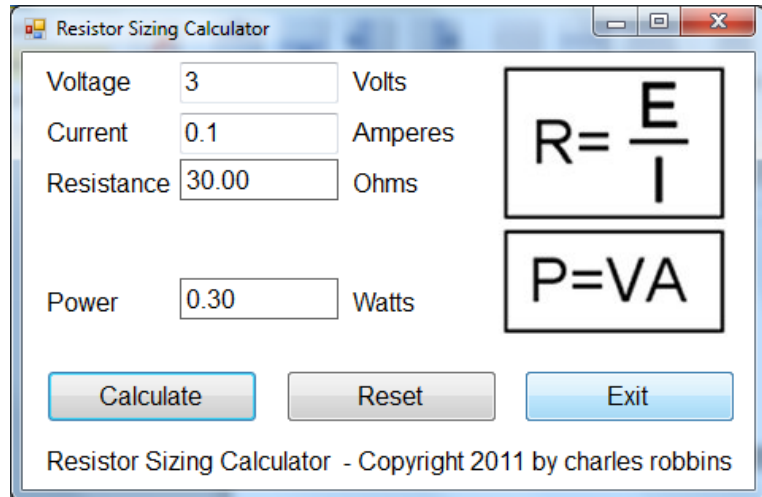


Figure 4.30 – Launching the Program

Type the voltage and current in the textbox just as we typed as shown in Figure 4.31. If we make a mistake, we can type over the text entry or press the Reset command button to clear the textbox. Press the Calculate command button and the two answer labels will have the resistance and power for the resistor. After experimenting with our program, press the Exit command button to exit the application.



**Figure 4.31 – Running the Program**

If our program does not function correctly, go back to the code and check the syntax against the program shown in previous sections. Repeat any processes to check or Beta test the program. When the program is working perfectly, save and close the project.

There are many variations of this Visual C# Application we can practice and obtain information from a personal computer. While we are practicing with forms, we can learn how to use variables, strings and comments. These are skills that we want to commit to memory.

**\* World Class CAD Challenge 90-4 \* - Write a Visual C# Application that displays a single input form, allows the user to type in their data, and when executed, the program will give the user information obtained from the computer and from mathematical computations.**

**Continue this drill four times using some other form designs, each time completing the Visual C# Project in less than 1 hour to maintain your World Class ranking.**