

2

The Visual C# Design Environment

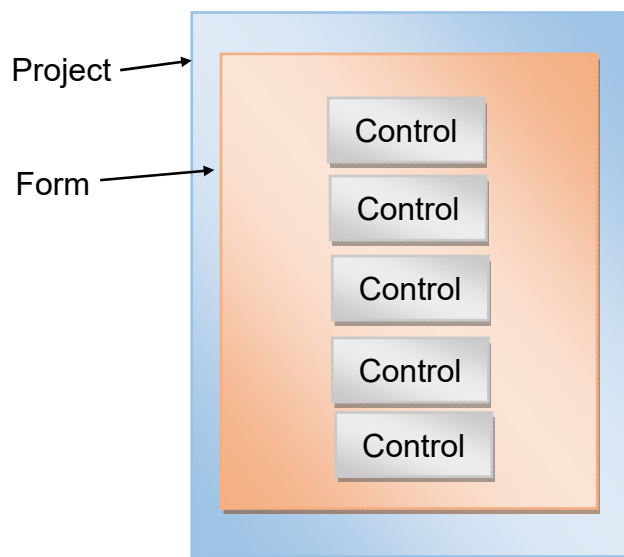


Review and Preview

In Class 1, we learned the important parts of a Visual C# **project**. We saw that a project is built on a **form** using **controls** (also called **objects**). By interacting with the controls using **events**, we get the computer to do assigned tasks via instructions we provide. In this second class, we will learn the beginning steps of building our own Visual C# projects by looking at the different parts of the project and where they fit in the Visual C# design environment. Like Class 1, there are also a lot of new terms and skills to learn.

Parts of a Visual C# Project

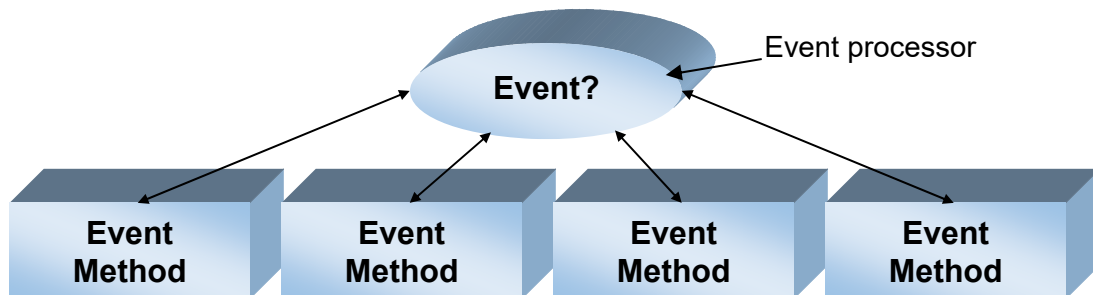
In Class 1, we saw that there are four major components in a Visual C# application: the solution, the project, the form, and the controls. A **solution** can contain multiple projects. In this course, solutions will only contain a single project, so the words solution and project are used interchangeably. **Project** is the word used to encompass everything in a Visual C# project. Other words used to describe a project are **application** or **program**. The **form** is the window where you create the interface between the user and the computer. **Controls** are graphical features or tools that are placed on forms to allow user interaction (text boxes, labels, scroll bars, command buttons). Recall the form itself is a control. Controls are also referred to as **objects**. Pictorially, a project is:



So, in simplest terms, a project consists of a form containing several (and some projects contain hundreds) controls.

Every characteristic of a control (including the form itself) is specified by a **property**. Example control properties include names, any text on the control, width, height, colors, position on the form, and contents. Properties are used to give your project the desired appearance. For each control studied in this class, we will spend a lot of time talking about properties.

In Class 1, we saw that by interacting with the controls in the **Sample** project (clicking buttons, choosing different options, typing text), we could make things happen in our project by generating control **events**. We say that Visual C# is an **event-driven** language and it is governed by an **event processor**. That means that nothing happens in a Visual C# project until some event occurs. Once an event is detected, the project finds a series of instructions related to that event, called an **event method**. That method is executed, then program control is returned to the event processor:



Event methods associated with various controls are where we do the actual computer programming. These methods are where we write C# language statements. You will learn a lot of programming and C# language in this class.

In summary, the major parts of a Visual C# project are:

- **form**
- **controls**
- control **properties**
- control **event methods**

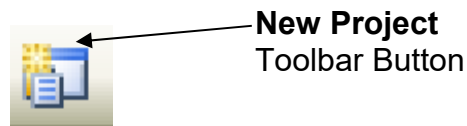
Now, let's take a look at the Visual C# programming environment and identify where we can access each of these project components.

Parts of the Visual C# Environment

Visual C# is more than just a computer language. It is a project building environment. Within this one environment, we can begin and build our project, run and test our project, eliminate errors (if any) in our project, and save our project for future use. With other computer languages, many times you need a separate text editor to write your program, something called a compiler to create the program, and then a different area to test your program. Visual C# integrates each step of the project building process into one environment. Let's look at the parts of the Visual C# environment. To help in this look, we first need to get a new project started. We won't do anything with this project. We just use it to identify parts of the Visual C# environment.

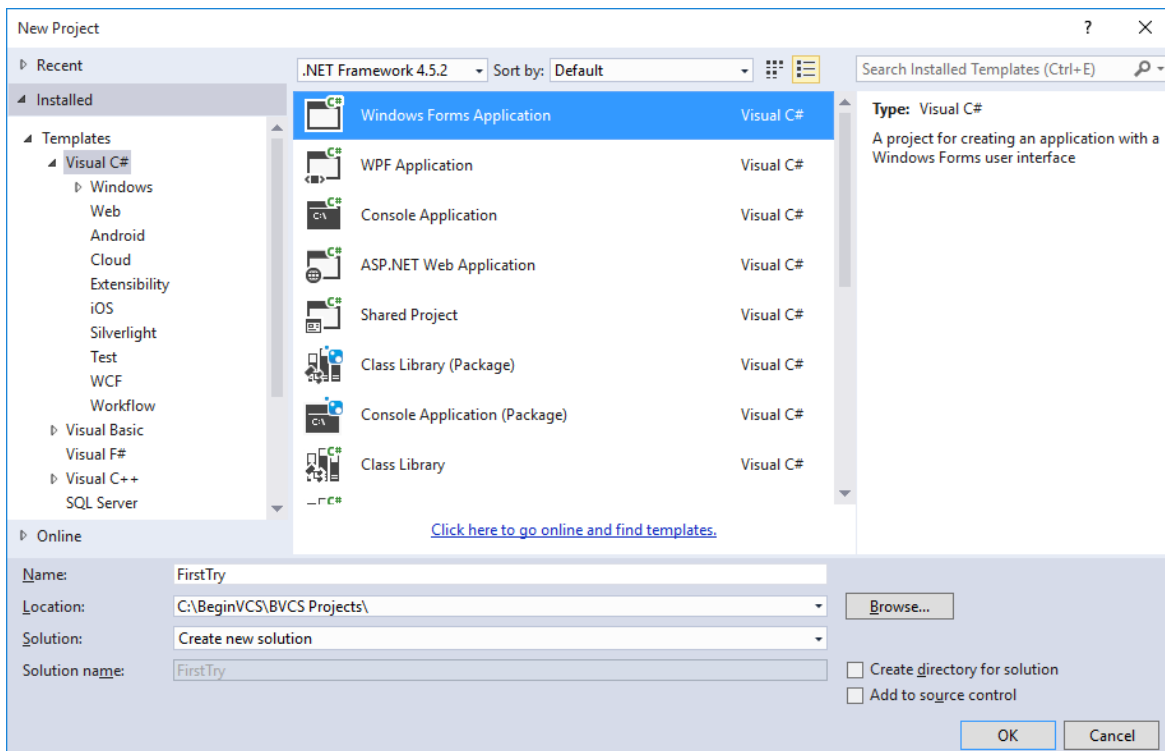
Starting a New Visual C# Project

Every time you want to build a project using Visual C#, a first step is to create a new project. Start Visual C# using the procedure learned in Class 1. We will start a new project using the toolbar under the Visual C# menu. Look for this button (the first button on the left):



You can also start a new project by selecting **File** from the menu, then clicking **New**, then **Project**.

Click the **New Project** button and a **New Project** box appears:

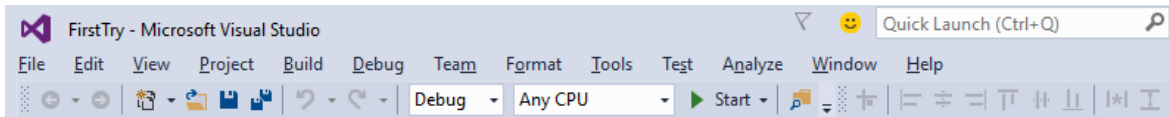


Under **Installed Templates**, make sure **Visual C#** is selected. We will always be building windows applications, so select **Windows Forms Application**.

This window also asks where you want to save your project. In the **Name** box, enter the name (I used **FirstTry**) of the folder to save your project in. **Location** should show the directory your project folder will be in. You can **Browse** to an existing location or create a new directory by checking the indicated box. For these notes, we suggest saving each of your project folders in the same directory. For the course notes, all project folders are saved in the **\BeginVCS\BVCS Projects** folder. Once done, click **OK**. Your new project will appear in the Visual C# environment, displaying several windows.

Main Window

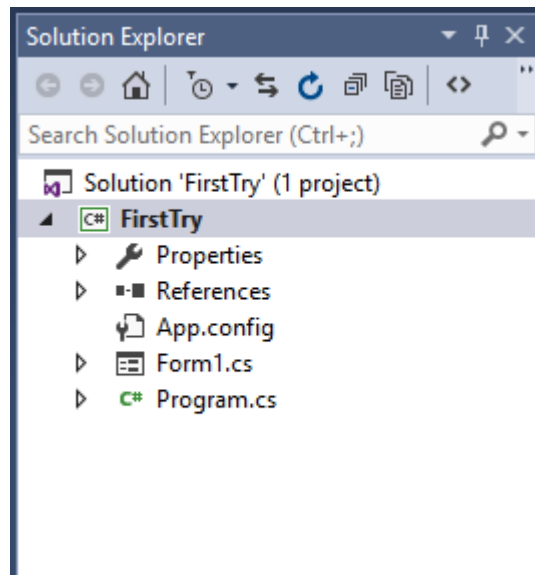
The **Main Window** is used to control most aspects of the Visual C# project building and running process:



The main window consists of the title bar, menu bar, and toolbars. The title bar indicates the project name (here, **FirstTry**). The menu bar has drop-down menus from which you control the operation of the Visual C# environment. The toolbars have buttons that provide shortcuts to some of the menu options. You should be able to identify the **New Project** button. Also, look for the button we used in Class 1 to start a project.

Solution Explorer Window

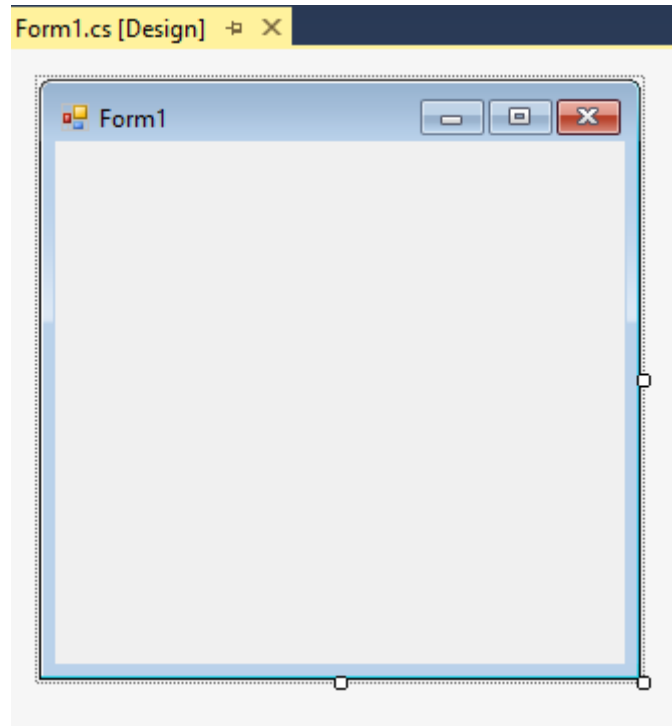
The **Solution Explorer Window** shows which files make up your project:



If the Solution Explorer window is not present on the screen, click **View** on the main menu, then **Solution Explorer**. If you select the form file (**Form1.cs**), you can obtain a view of the project form by choosing the **View** menu, then **Designer**. Or, you see the actual C# coding within a form by clicking the **View Code** button in the **Solution Explorer** window. We will look at this code window soon.

Design Window

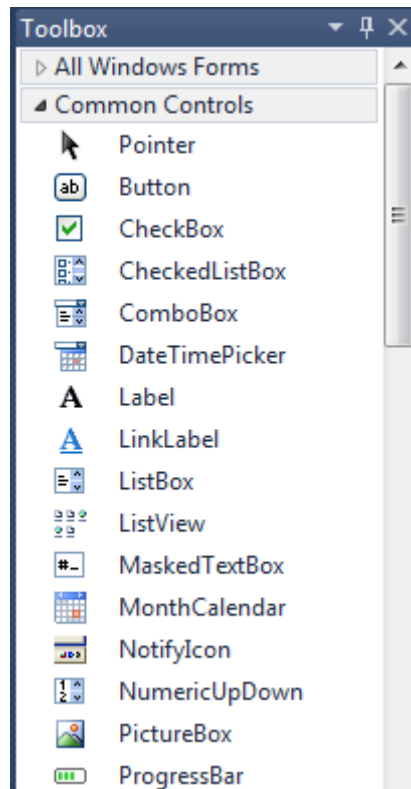
The **Design Window** is central to developing Visual C# applications. It is where you build your form and write actual code. You should see a blank form in this window:



If the form is not present on the screen, select **Form1.cs** in the **Solution Explorer** window. Then, click **View** on the main menu, then **Designer**. Or, press the **<F7>** function key while holding down **<Shift>**.

Toolbox Window

The **Toolbox Window** is the selection menu for controls used in your application. Many times, controls are also referred to as **objects** or **tools**. So, three words are used to describe controls: objects, tools, and, most commonly, controls.

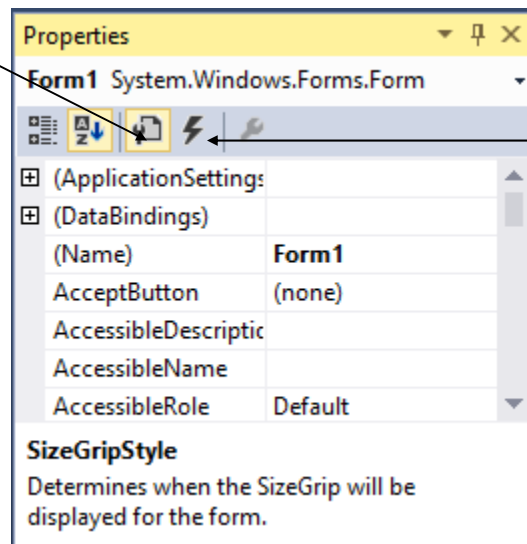


If the toolbox window is not present on the screen, click **View** on the main menu, then **Toolbox**. Make sure you are viewing the **Common Controls**. See if you can identify some of the controls we used in Class 1 with our **Sample** project.

Properties Window

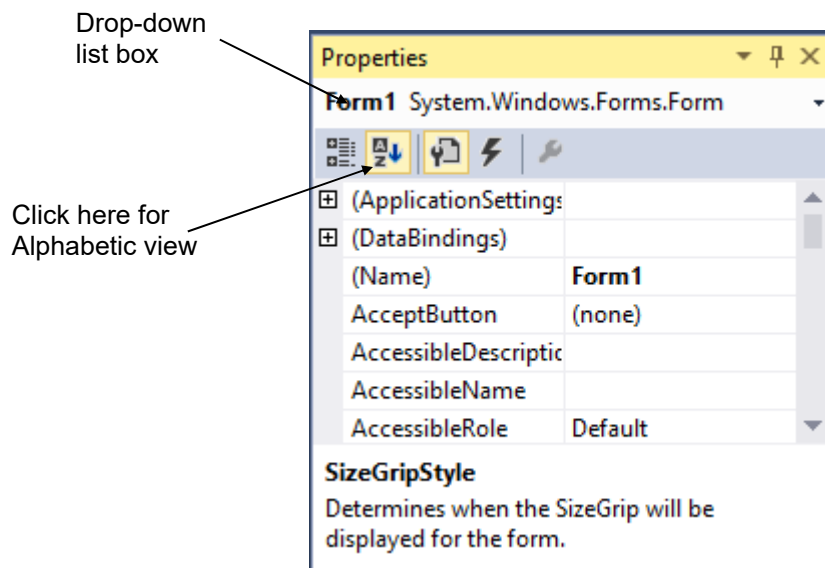
The **Properties Window** is used to establish initial property values for controls. It is also used to establish control **events** (we will see how in Class 3) – for now, we just look at the properties – to do this make sure the **Properties** toolbar button (in the properties window) is selected and not the **Events** button (see the picture below):

Click here to
see **Properties**



Click here to
see **Events**

The drop-down box at the top of the window lists all controls on the current form. Under this box are the available properties for the currently selected object (the **Form** in this case). Different views of the properties are selected using the toolbar near the top of the window. Two views are available: **Alphabetic** and **Categorized**. We will always use the Alphabetic view.



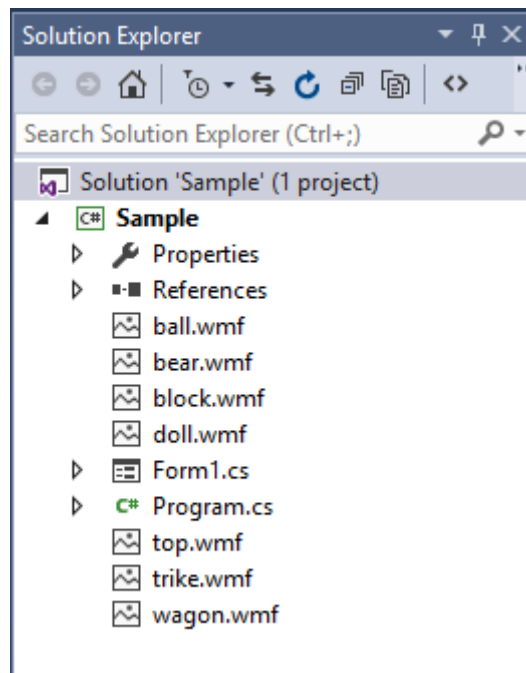
If the properties window is not present on the screen, click **View** on the main menu, then **Properties Window**. As an alternate, if the window does not show up, press the **F4** function key. Note the properties window will only display when the form and any controls are displayed in the Design window.

You should be familiar with each of the Visual C# environment windows and know where they are and how to locate them, if they are not displayed. Next, we'll revisit the project we used in Class 1 to illustrate some of the points we've covered here.

Moving Around in Visual C#

Solution Explorer Window

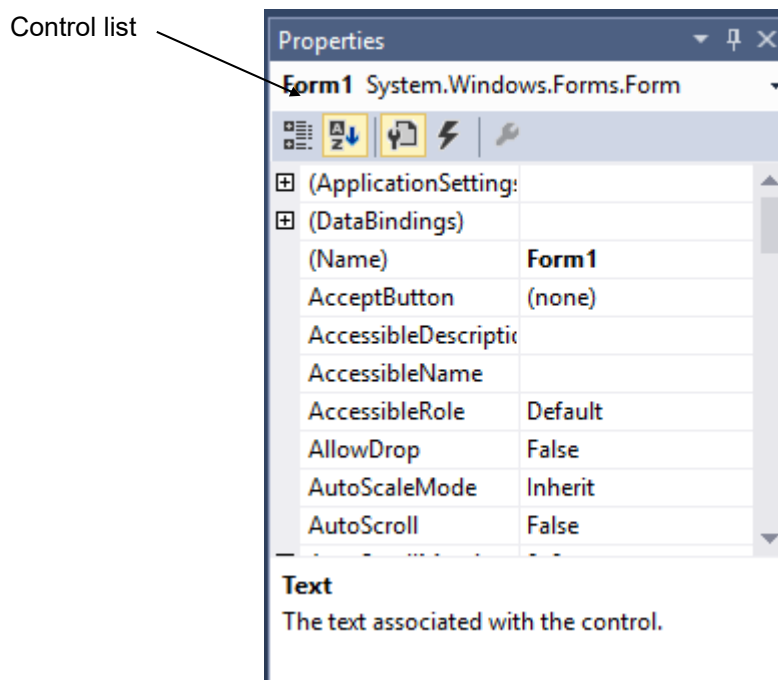
Open the project named **Sample** that we used in Class 1 (use the **File** menu option, then select **Open** and **Open Project** reviewing the steps in Class 1 if needed). Once **Sample** is opened (recall it is in the **Sample** folder in the **\BeginVCS\BVCS Projects** folder), find and examine the **Solution Explorer** window:



The Solution Explorer window indicates we have a solution with a project file named **Sample**. The project contains a single form saved as **Form1.cs**. The project also includes folders named **Properties** and **References** and a file named **Program.cs**. There are also several graphics files (the ones with **wmf** extensions). The only file we're really worried about for now is the form.

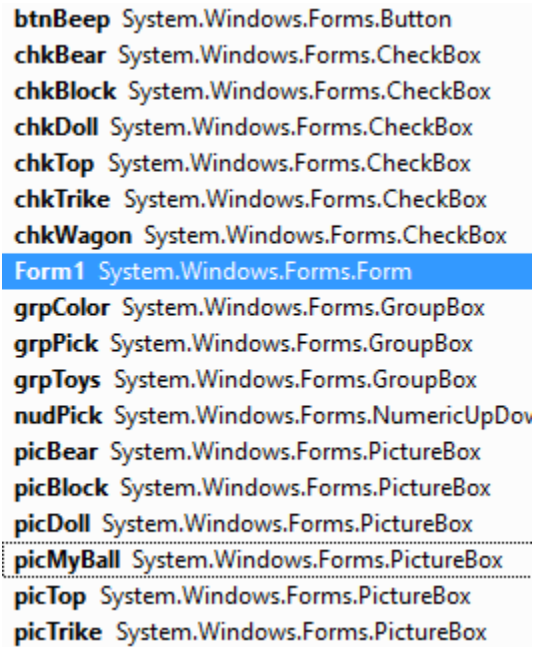
Properties Window

Find the **Properties** window. Remember it can only be shown when the form is displayed. So, you may have to make sure the form is displayed first. Review the steps that get the desired windows on your screen. Make sure the properties and not the events are displayed.



The drop-down box at the top of the properties window is called the **control list**. It displays the name (the **Name** property) of each control used in the project, as well as the type of control it is. Notice, as displayed, the current control is the **Form** and it is named **Form1**. The properties list is directly below this box. In this list, you can scroll (using the scroll bar) through the properties for the selected control. The property name is on the left side of the list and the current property value is on the right side. Scroll through the properties for the form. Do you see how many properties there are? You'll learn about many of these as you continue through the course. Don't worry about them for now, though.

Click on the down arrow in the control list (remember that's the drop-down box at the top of the properties window):

A screenshot of the Visual Studio control list dropdown menu. The list contains various Windows Forms controls and their namespaces. The controls are listed in two columns. The first column contains: btnBeep, chkBear, chkBlock, chkDoll, chkTop, chkTrike, chkWagon, Form1 (highlighted in blue), grpColor, grpPick, grpToys, nudPick, picBear, picBlock, picDoll, picMyBall (highlighted with a dotted border), picTop, and picTrike. The second column contains the corresponding namespaces: System.Windows.Forms.Button, System.Windows.Forms.CheckBox, System.Windows.Forms.CheckBox, System.Windows.Forms.CheckBox, System.Windows.Forms.CheckBox, System.Windows.Forms.CheckBox, System.Windows.Forms.CheckBox, System.Windows.Forms.Form, System.Windows.Forms.GroupBox, System.Windows.Forms.GroupBox, System.Windows.Forms.GroupBox, System.Windows.Forms.NumericUpDown, System.Windows.Forms.PictureBox, System.Windows.Forms.PictureBox, System.Windows.Forms.PictureBox, System.Windows.Forms.PictureBox, System.Windows.Forms.PictureBox, and System.Windows.Forms.PictureBox.

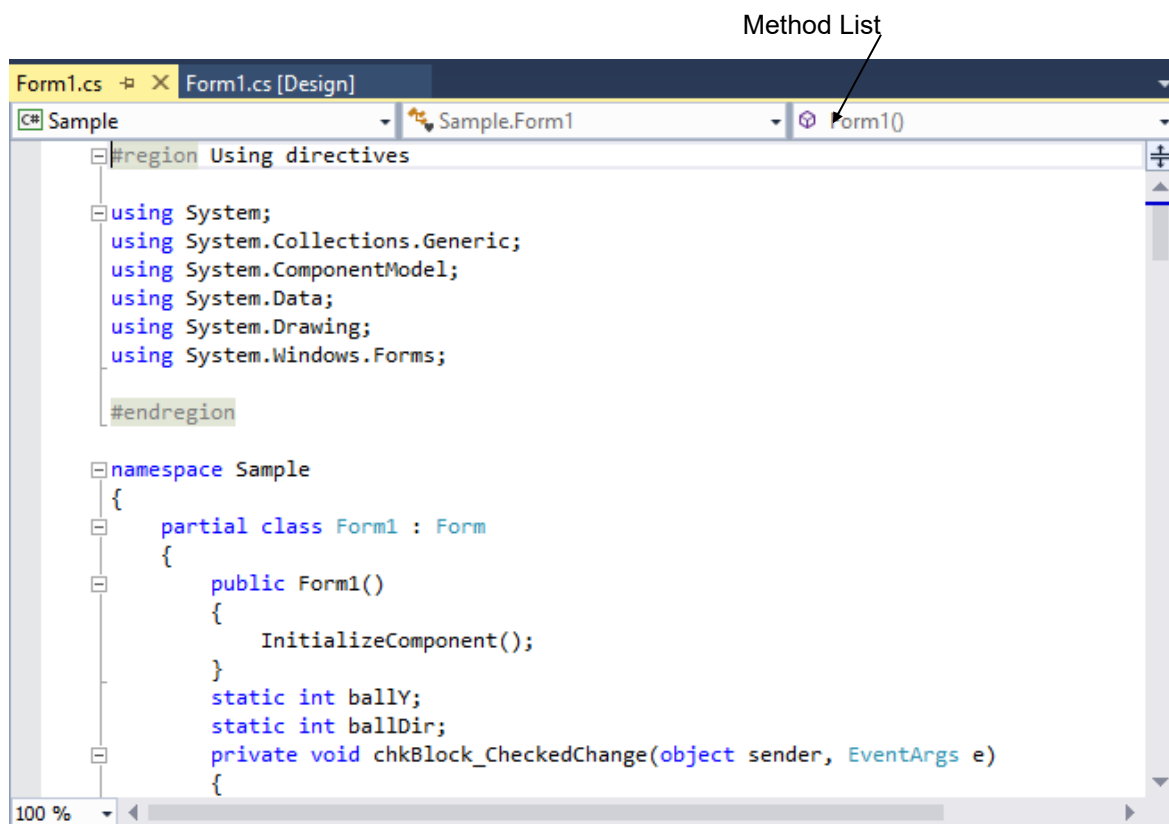
```
btnBeep System.Windows.Forms.Button
chkBear System.Windows.Forms.CheckBox
chkBlock System.Windows.Forms.CheckBox
chkDoll System.Windows.Forms.CheckBox
chkTop System.Windows.Forms.CheckBox
chkTrike System.Windows.Forms.CheckBox
chkWagon System.Windows.Forms.CheckBox
Form1 System.Windows.Forms.Form
grpColor System.Windows.Forms.GroupBox
grpPick System.Windows.Forms.GroupBox
grpToys System.Windows.Forms.GroupBox
nudPick System.Windows.Forms.NumericUpDov
picBear System.Windows.Forms.PictureBox
picBlock System.Windows.Forms.PictureBox
picDoll System.Windows.Forms.PictureBox
picMyBall System.Windows.Forms.PictureBox
picTop System.Windows.Forms.PictureBox
picTrike System.Windows.Forms.PictureBox
```

Scroll through the displayed list of all the controls on the form. There are a lot of them. Notice the assigned names and control types. Notice it's pretty easy to identify which control the name refers too. For example, **picBear** is obviously the **picture box** control holding a picture of a bear. We always want to use proper control naming - making it easy to identify a control just by its name. We'll spend time talking about control naming in the later classes.

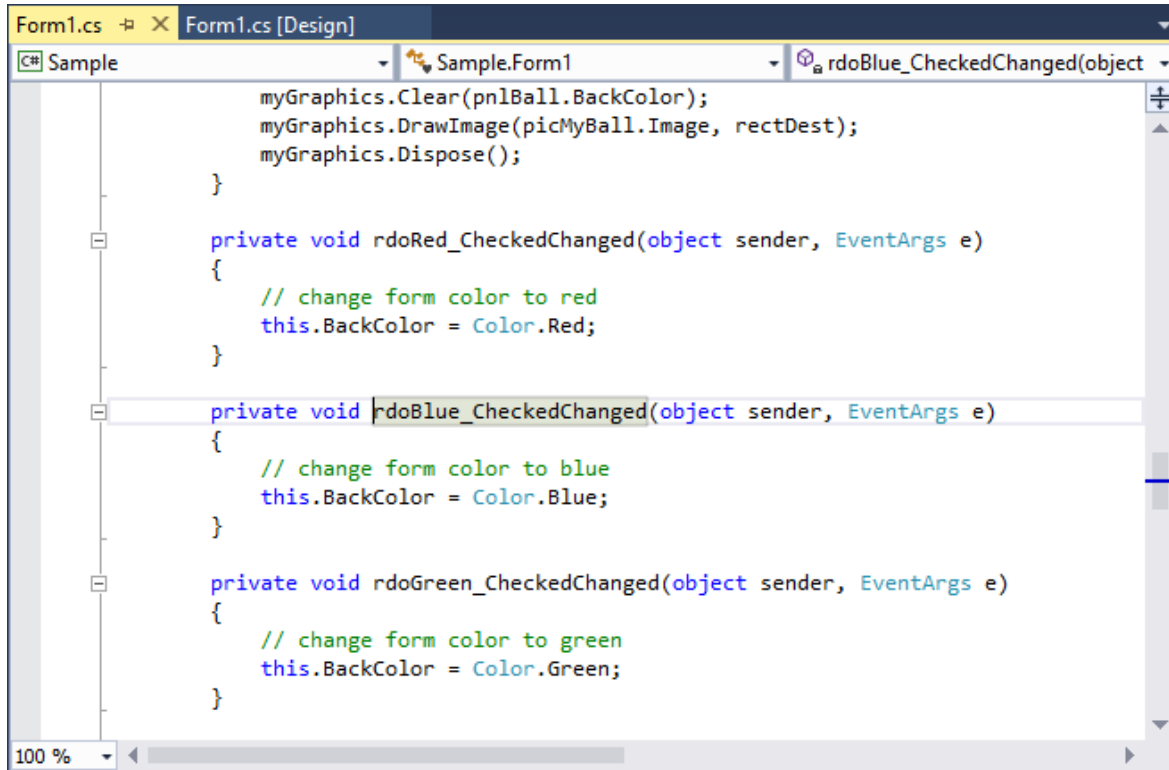
Select a control and scroll through the properties for that control. Look at the properties for several controls. Notice every control has many properties. Most properties are assigned by default, that is the values are given to it by Visual C#. We will change some properties from their default values to customize them for our use. We will look at how to change properties in Class 3.

Code Window

Let's look at a new window. Recall Visual C# is event-driven - when an event is detected, the project goes to the correct **event method**. Event methods are used to tell the computer what to do in response to an event. They are where the actual computer programming (using the C# language) occurs. We view the event methods in the **Code Window**. There are many ways to display the code window. One way is to use the **View Code** button found in the Solution Explorer window. Another is to click **View** on the main menu, then **Code**. Or, as an alternate, press the **F7** function key. Find the code window for the **Sample** project. It will appear in the design window under the **Form1.cs** tab:



At the top of the code window are two drop-down boxes. The one on the right side is the **method lists**. It lists all the methods (including event methods) used in the code. Click on the drop-down arrow in the methods list. Select **rdoBlue_CheckedChanged** as the method. You should see this:



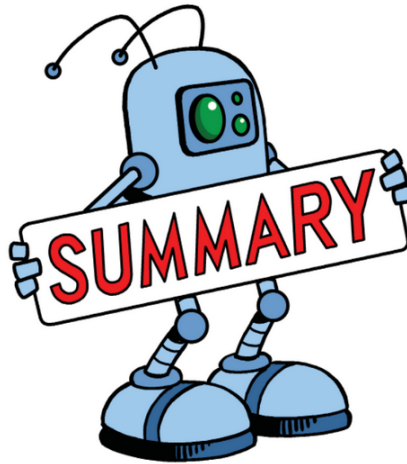
Near the top of the code window is the **CheckedChanged** event method for the control name **rdoBlue**. And even though you may not know any C# right now, you should be able to figure out what is going on here. Since we will be careful in how we name controls, you should recognize this control to be the radio button (one with a little circle) with the word **Blue** next to it (the word next to a radio button is its **Text** property). The status of a radio button (whether it is selected or not) is called its **Checked** property. So, this event method is called whenever we click on the Blue radio button and change its Checked property.

Notice the procedure has a single line of instruction (ignore the other lines for now):

```
this.BackColor = Color.Blue;
```

What this line of C# code says is set the **BackColor** property of the control named **this** (a word used by Visual C# to refer to the form) to Blue (represented by the words **Color.Blue**). Pretty easy, huh?

Scroll through the other code in the code window. Much of this code might look like a foreign language right now and don't worry - it should! You'll be surprised though that you probably can figure out what's going on even if you don't know any C#. In subsequent classes, you will start to learn C# and such code will become easy to read. You'll see that most C# code is pretty easy to understand. Writing C# code is primarily paying attention to lots of details. For the most part, it's very logical and obvious. And, you're about to start writing your own code!



In this second class, we've learned the parts of the Visual C# environment and how to move around in that environment. We've also learned some important new terms like **properties** and **event methods**. You're now ready to build your first Visual C# project. In the next class, you'll learn how to place controls on a form, move them around, and make them appear just like you want. And, you will learn the all-important step of how to put C# code in the event methods.