1

Introducing Visual C#

A Brief History of Visual C#

In the mid-1960's, most computing was done on large computers taking up entire floors of buildings (these machines had less computational power than the laptop I'm typing these notes on!).

Most programming was done in cryptic languages by engineers and mathematicians. Two professors at Dartmouth College wanted to explain programming to "normal" people and developed the **BASIC** (Beginner's All-Purpose Symbolic Code) language to help in that endeavor. BASIC was meant to be a simple language with just a few keywords to allow a little math and a little printing.

In the later 1960's, timeshare computing, where a user could sit at a terminal and interact with the computer, became popular. The primary language used in these interactive sessions was BASIC. The Dartmouth BASIC was not sufficient for the many applications being developed, so many extensions and improvements were made in the BASIC language. Many of the first computer games were written on

timeshare terminals using BASIC – gambling games, world simulations and the classic Star Trek game were very popular.

In the mid-1970's, an issue of Popular Science magazine changed the world of computers forever. On the cover was an Altair computer. About all the computer could do was flash some lights according to a program written by the user. But, it was the first home computer. Two young guys in Seattle, Bill Gates and Paul Allen, saw the potential. They developed a BASIC language for the Altair computer and marketed it through their new company – Microsoft. It sold for $350 and was distributed on a cassette tape.

Since those early days, the folks at Microsoft have developed many other products and many other programming languages. The product you will learn in this set of notes is called **Visual C#**. The word **Visual** means you will build Windows-based applications that a user can see and interact with. The term **C#** (pronounced "cee sharp") refers to the particular language used within the Visual C# environment. This language was developed using pieces of other languages called C, C++ and Java. Visual C# is one of the easiest programming languages to learn. Yet, even though it is easy to learn and to use, Visual C# can also be used to develop very powerful computer programs. Visual C# provides a sophisticated environment for building and testing Windows-based applications. You've used Windows-based applications before. Microsoft's programs like Word, Excel, Internet Explorer and the windows that appear within these applications (to open and save files, to print files) are all Windows-based applications. These applications are not written in Visual C# (they are written in a language called C++), but they do demonstrate the functionality you can put in your Visual C# applications.

Visual C# can be used to write computer games, businesses can use Visual C# to manage their databases, webmasters can use Visual C# to develop web pages, and people like yourself can use Visual C# to build Windows applications they want and need in their everyday home and work life. In these notes, you will learn how to use Microsoft's Visual C# to write your own Windows-based applications. You may not become a billionaire like Bill and Paul, but hopefully you'll have some fun learning a very valuable skill.

Let's Get Started

Learning how to use Visual C# to write a computer program (like learning anything new) involves many steps, many new terms, and many new skills. We will take it slow, describing each step, term, and skill in detail. Before starting, we assume you know how to do a few things:

- You should know how to start your computer and use the mouse.
- You should have a little knowledge on working with your operating system.
- You should know how to resize and move windows around on the screen.
- You should know how to run an application on your computer by using the **Start Menu**.
- You should know how to fill in information in Windows that may pop up on the screen.
- You should know about folders and files and how to find them on your computer.
- You should know what file extensions are and how to identify them. For example, in a file named **Example.ext**, the three letters **ext** are called the extension.
- You should know how to click on links to read documents and move from page to page in such documents. You do this all the time when you use the Internet.

You have probably used all of these skills if you've ever used a word processor, spreadsheet, or any other software on your computer. If you think you lack any of these skills, ask someone for help. They should be able to show you how to do them in just a few minutes. Actually, any time you feel stuck while trying to learn this material, never be afraid to ask someone for help. We were all beginners at one time and people really like helping you learn.

Let's get going. And, as we said, we're going to take it slow. In this first class, we will learn how to get Visual C# started on a computer, how to load a program (or project) into Visual C#, how to run the program, how to stop the program, and how to exit from Visual C#. It will be a good introduction to the many new things we will learn in the classes to come.

Starting Visual C#

We assume you have Visual C# installed and operational on your computer. If you don't, you need to do this first. Again, this might be a good place to ask for someone's help if you need it. Visual C# is available for free download from Microsoft.

Visual C# is included as a part of **Microsoft Visual Studio 2015 Community Edition**. Visual Studio includes not only Visual C#, but also Visual C++ Express and Visual Basic Express. All three languages use the same development environment. Follow this link for complete instructions for downloading and installing Visual C# on your computer:

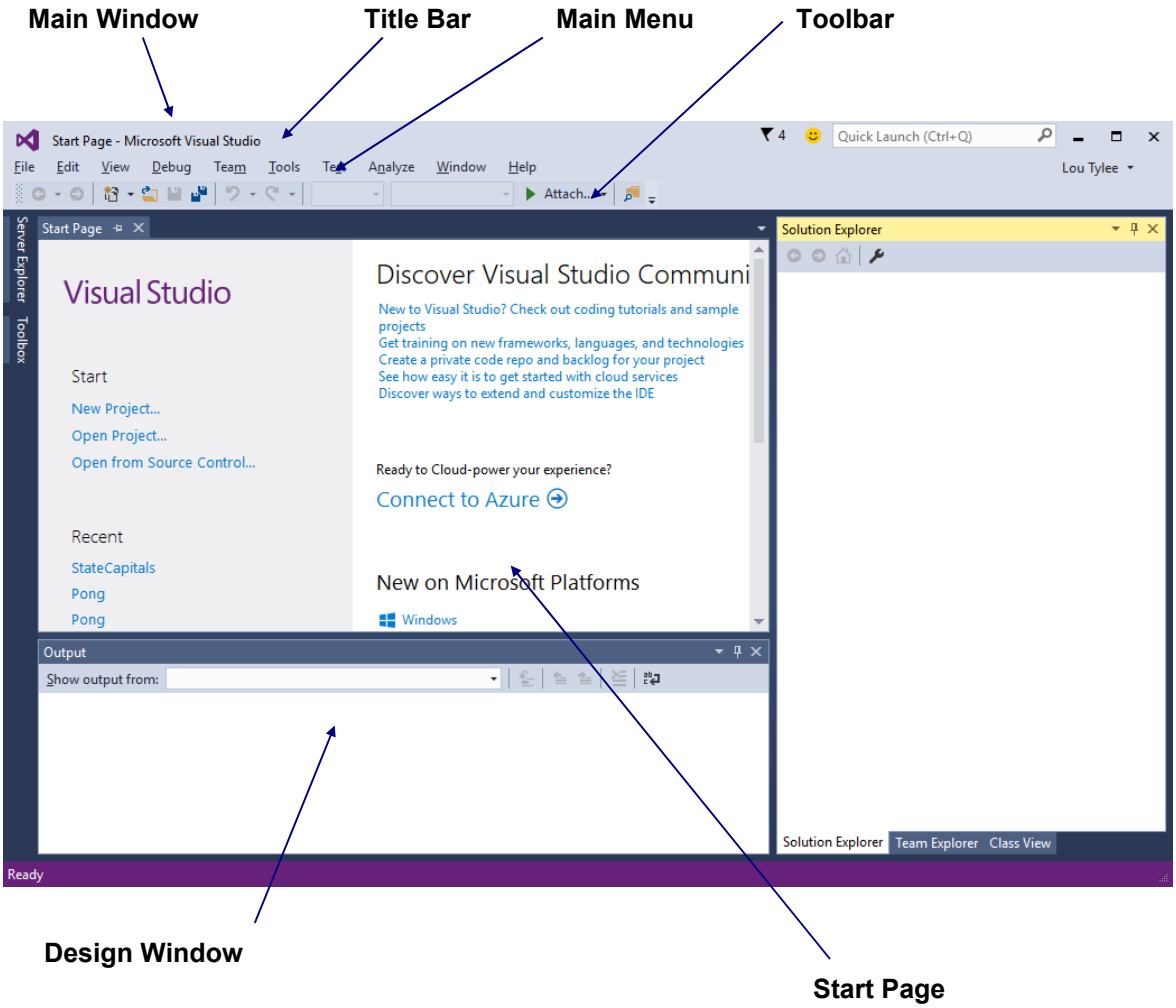
https://www.visualstudio.com/products/free-developer-offers-vs

Once installed, to start Visual C#:

- Click on the **Start** button on the Windows task bar.
- Click **All apps**
- Then select **Microsoft Visual Studio 2015**

The Visual C# program should start. Several windows will appear on the screen, with the layout depending on settings within your product.

Upon starting, my screen shows:



This screen displays the Visual C# **Integrated Development Environment (IDE)**. This is where we build, run and work with our applications. Let's point out just a few items on the screen. There are many windows on the screen. At the top of the screen is the Visual C# **Main Window**. At the top of the main window is the **Title Bar**. The title bar gives us information about what program we're using and what Visual C# program we are working with. Below the title bar is the **Main Menu** from where we can control the Visual C# program. You should be familiar with how menus work from using other programs like word processors and games. Under the main menu is a **Toolbar**. Here, little buttons with pictures also allow us to control Visual C#, much like the main menu. If you put the mouse cursor over one of these buttons for a second or so, a little 'tooltip' will pop up and tell you

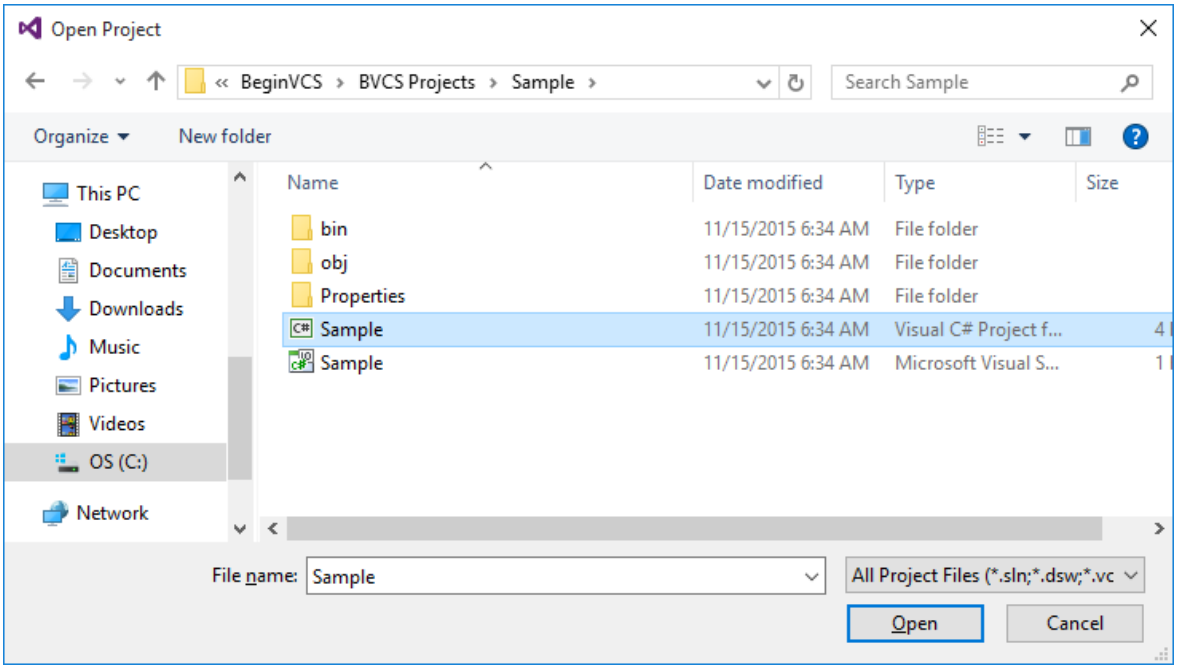
what that particular button does - try it! Almost all Windows applications (spreadsheets, word processors, games) have toolbars that help us do different tasks. This is the purpose of the Visual C# toolbar. It will help us do most of our tasks. In the middle of the screen is the **Start Page**, contained in the **Design Window**. This page has many helpful topics you might be interested in pursuing as you learn more about Visual C#. – especially note the topics under **Discover Visual Studio Community 2015**.

At any time, your particular screen may look different than ours. The Visual C# environment can be customized to an infinite number of possibilities. This means you can make things look anyway you want them to. You can 'dock' windows or 'float' windows. You can move windows wherever you want or you can completely delete windows. And, different windows will appear at different times. As you become more experienced with Visual C#, you will learn ways you want things to be. We encourage you to try different things. Try moving windows. Try docking and floating. We won't talk a lot about how to customize the development environment. (We will, however, always show you how to find the particular window you need.)

Opening a Visual C# Project

What we want to do right now is **open a project**. Windows applications written using Visual C# are referred to as **solutions**. A solution is made up of one or more **projects**. Projects include all the information we need for our computer program. In this course, our applications (solutions) will be made up of a single project. Because of this, we will use the terms application, solution and project interchangeably. Included with these notes are many Visual C# projects you can open and use. Let's open one now.

- Select **File** from the main menu, then click **Open,** then **Project/Solution**. An **Open Project** window will appear:

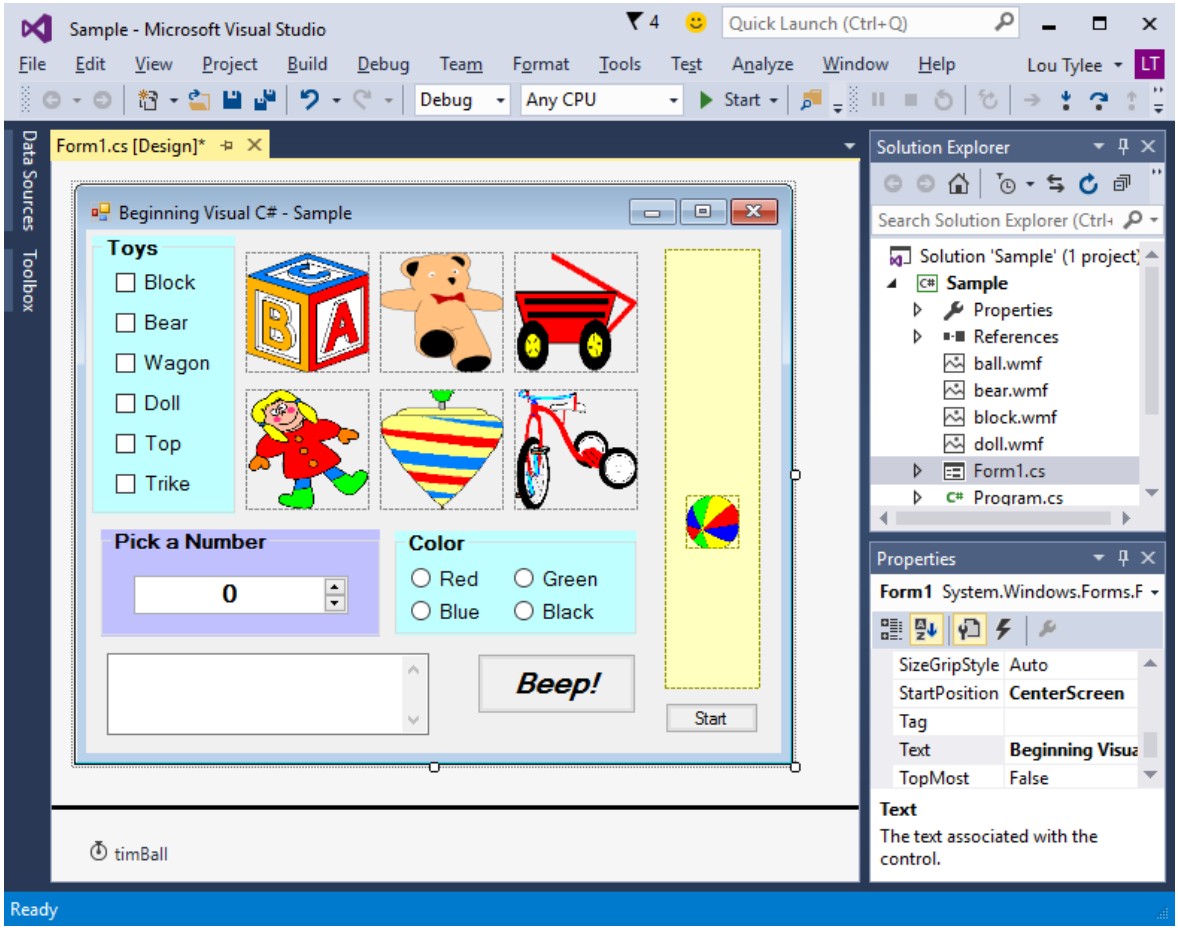


- Find the folder named **BeginVCS** (stands for **Beginning Visual C# (Sharp)**). This is the folder that holds the notes and projects for this course. Open that folder.
- Find and open the folder named **BVCS Projects.** This folder holds all the projects for the course

Remember how you got to this folder. Throughout the course, you will go to this folder to open projects you will need. Open the project folder named **Sample**.

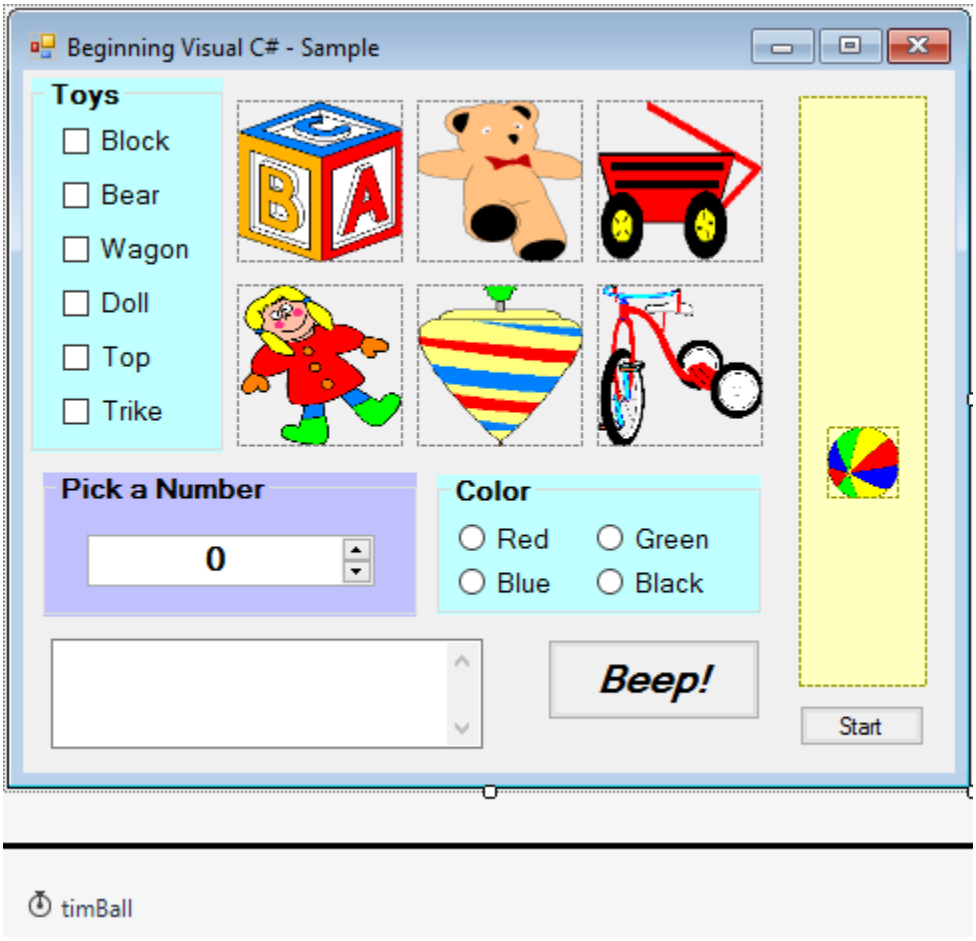
In this project folder, among other things is a **Visual Studio Solution** file named **Sample** (with **sln** extension) and a **Visual C# Project** file named **Sample** (with **csproj** extension). Open the **Sample** solution file (as shown in the example Open Project window). Since there is only one project in this solution, you could also open the project file and get the same results, but it is better to always open the solution file.

Once the project is opened, many windows are now on the screen:



Look for the **Solution Explorer** window (if it is not there, choose **View** in the menu and select **Solution Explorer**). This lists the files in our solution. Right-click the file **Form1.cs** and choose **Open**.

In the **Design** window will appear a window that looks something like this:



This is our project named **Sample**. We're going to spend a bit of time explaining everything that is displayed here. This will introduce you to some of the words, or vocabulary, we use in Visual C#. There are lots of terms used in Visual C#. Don't try to memorize everything - you'll see these new words many times through the course.

We call the displayed project window a **Form**. All Visual C# projects or programs are built using forms. In fact, you have probably noticed that all Windows applications are built using forms of some type. At the top of the form is the **Title Bar**. It has an **icon** (little picture) related to the form, a description of what the form does (**Beginning Visual C# - Sample**), and three smaller buttons that control

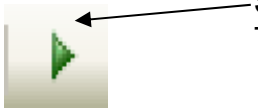
form appearance (we won't worry about these buttons right now). There are lots of other things on the form. These other things are the 'heart' of a Visual C# computer program.

You see a set of square buttons with toy names next to them. You see pictures of toys. You see a set of round buttons with color names next to them. There is a little box you can type in with something called a scroll bar on the right side. There's a big button that says **Beep!** There's a little device for picking the value of a number. And, there's a ball in a big rectangle with a button that says **Start** and, below the form, a little thing that looks like a stopwatch. We call all of these other things on the form **Controls** or **Objects**. Controls provide an **interface**, or line of communication, between you (or the user of your program) and the computer. You use the controls to tell the computer certain things. The computer then uses what it is told to determine some results and displays those results back to you through controls. By the way, the form itself is a control. If you've used any Windows applications, you've seen controls before - you probably just didn't know they were called controls. As examples, buttons on toolbars are controls, scroll bars to move through word processor documents are controls, menu items are controls, and the buttons you click on when opening and saving files are controls.

I think you get the idea that controls are a very important part of Visual C#, and you're right. They are the most important part of Visual C# - they are what allow you to build your applications. We will spend much of this course just learning about controls. Right now, though, let's run this program and get some insight into how a Visual C# project (and its controls) works.

Running a Visual C# Project

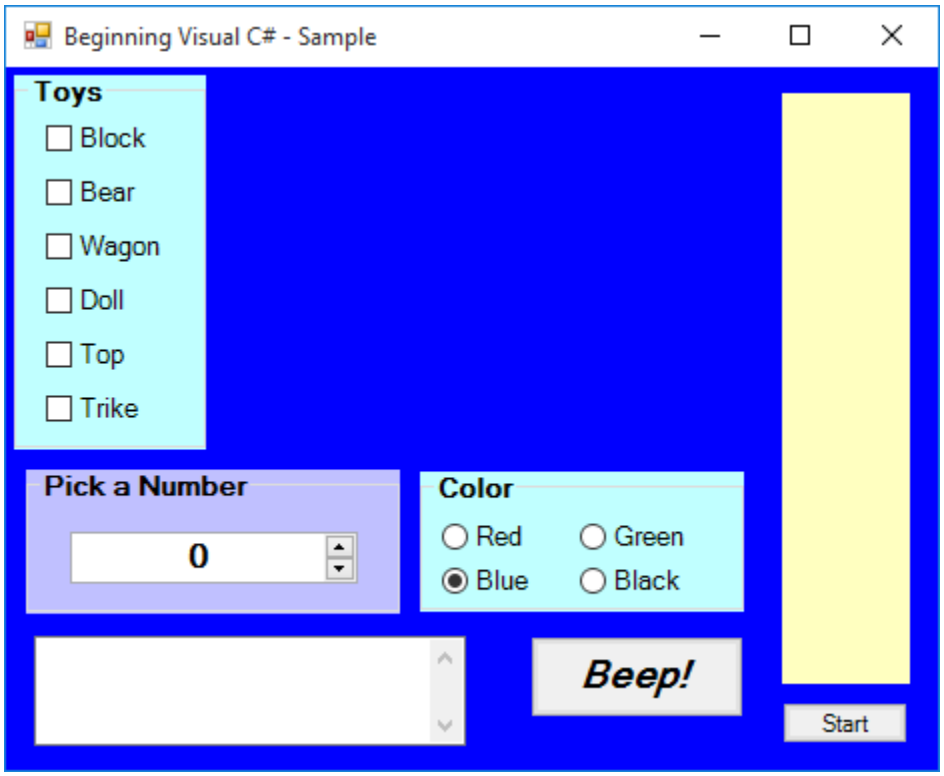
After developing a Visual C# project, you want to start or run the program. This gets the program going and lets the user interact with the **controls** on the form and have the computer do its assigned tasks. We can run a project using the toolbar under the Visual C# menu. Look for a button that looks like the **Play** button on a VCR, CD player, or cassette tape player:



Click this button to run **Sample** (the project we opened previously).

You can also run a project by: (1) selecting the **Debug** menu heading, then clicking **Start Debugging**, or (2) pressing the **<F5>** function key.

The project form will appear and look something like this. Your form may appear slightly different depending on the particular Windows operating system you are using. We use both Windows Vista (seen here) and Windows XP in these notes:



Notice a few things have changed. All the toys have disappeared. The background color of the form is blue. The circle button next to **Blue** has a black dot in it. The little stopwatch control is not visible. The little ball has moved near the top of the big rectangle. What happened? We'll find out how and why all this happened as we learn more about Visual C#. Also, notice in the Visual C# title bar (in the main window) that the word **Running** appears in parentheses next to the project name. It is important to always know if you are running (in run mode) or designing a program (in design mode) – this indication in the title bar will tell you.

The project is now running, but what is it doing? Nothing is happening, or is it? At this point, Visual C# is waiting for you, the user, to do something. We say your Visual C# project is waiting for an **event** to occur. Nothing can happen in a Visual

C# program until an event occurs. We call Visual C# an **event-driven** programming language. So, let's cause an event.

An event occurs when you do something on the form - click on something with the mouse, type something in places where words can go, or maybe drag an object across the form. In the upper left corner of the form is a group of six boxes within a rectangular region with the heading **Toys**. Each little box has a toy name printed next to it. Click on one of these boxes. Notice what happens. A check appears in the selected box, indicating box selection, and the toy named by that box appears on the screen. When we click on a box, we cause an event, called a **CheckedChanged** event (this means the 'checked' status of the box has changed). The computer recognizes the event and does what you have told it to do (through your computer program) if that particular event occurs. In this case, the event tells the computer to display the selected toy. Click on the box again. The check mark and the toy disappear. You have caused another event and told the computer to make the toy disappear. This particular control is called a **check box**. Notice you can check as many boxes as you want, picking which toys (if any) you want displayed on your screen. Check boxes are used when you want to select items from a list. Two other controls are used in this example. The rectangular region the check boxes are contained is called a **group box**. The region each toy picture is displayed in is called a **picture box** control. Now, let's look at causing events with the other controls on the form.

Near the middle of the screen is a group of four round buttons in a group box with the heading **Color**. Each button has a color name printed next to it. The **Blue** button has a black dot in it, indicating it is the currently selected color (notice the form is blue). Click on another of these buttons. Notice what happens. The form color changes to the selected color. This **CheckedChanged** (meaning the 'checked' or actually 'dotted' status of the button has changed) event tells the computer to change the form background color. Notice that when you select a new color, the black dot appears in the selected button and disappears in the

previously selected button. Unlike the check boxes we saw earlier, you can only select one of these buttons. This makes sense - the form can only be one color! These round buttons are called **radio buttons**. Radio buttons are used when you need to choose exactly one option from a list of many. They are called radio buttons because, on a radio, you can only choose one option (station) at a time.

Under the **Toys** group box is another group box with the heading **Pick a Number**. There we see a control called a **numeric up-down** control. There is a **label** area displaying a number and next to the number is another control with one arrow pointing up and one pointing down (a **scroll bar**). You've probably seen scroll bars in other applications you have used. The scroll bar is used to change the displayed number. Click on the arrow on the top of the scroll bar. The displayed value will increase by 1. Continued clicking on that arrow will continue to increase the value. Clicking the lower arrow will decrease the value. In this example, the computer is responding to the numeric up-down control's **ValueChanged** event, which occurs each time an arrow is clicked, changing the displayed value.

Under the **Pick a Number** group box is a region with a scroll bar on the right side. This control is called a **text box**. You can click in it, then type in any text you want. Try it. The text box is like a little word processor in itself. Each time you type something in the text box, several events occur. There is a **KeyPress** event when you press a key and a **Change** event that is called each time the text in the box changes.

Next to the text box is a button that says **Beep!** Click the button and you should hear a beep on your computer's speaker. This control is called a **button** and is one of the most widely used controls in Visual C#. The **Click** event told the computer to make the speaker beep.

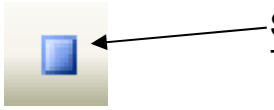
The last thing on our form is a tall, yellow, rectangular control called a **panel** that contains a **picture box** control displaying a beach ball. Under the panel is a

button that says **Start**. Click on that button, that is, cause a **Click** event. The ball starts moving down. It continues moving down until it hits the bottom of the panel, then starts moving back up. It will continue to do this until you click the button that now says **Stop**. Remember the little stopwatch that was below our form in design mode, but disappeared when we ran the project. It is being used by the bouncing ball example - it is called a **timer** control. The Click event on the button, in addition to changing what the button says to **Stop**, also started this timer control. The timer control generates **Tick** events all by itself at preset time intervals. In this example, a **Tick** event is generated every 1/10th of a second and, in that event, the ball position is changed to give the appearance of movement. Notice that even while the ball is bouncing, you can change the form color, make toys appear and disappear, type text, and make the computer beep. So, Visual C# even has the capability of handling multiple events.

Obviously, this project doesn't do much more than demonstrate what can be done with Visual C#, but that is a important concept. It points out what you will be doing in building your own Visual C# projects. A project is made up of the controls that let the user provide information to the computer. By causing events with these controls, the computer will generate any required results. We haven't worried about how to use the events to determine these results, but we will in all the later classes. By the time you have finished this course, you will be able to build projects that do everything (and more) that the **Sample** project does. Let's look now at how to stop the project.

Stopping a Visual C# Project

There are many ways to stop a Visual C# project. We will use the toolbar. Look for a button that looks like the **Stop** button on a VCR, CD player, or cassette tape player (you may have to move the project form down a bit on the screen to see the toolbar):



Click on this button (you may have to click it twice). The project will stop and Visual C# will return to design mode.

Alternate ways to stop a project are:

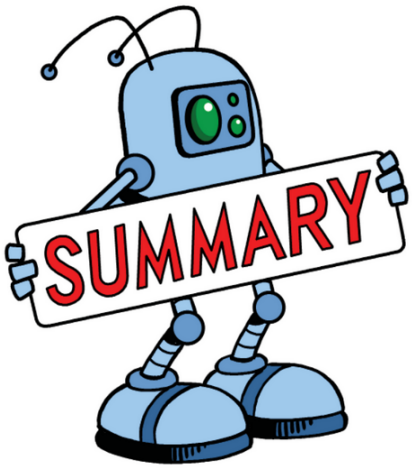
- Selecting the **Debug** menu heading, then clicking **Stop Debugging**
- Click the **Close** button found on the form. It is the little button that looks like an **X** in the upper right corner of the form.

Stopping Visual C#

When you are done working with a Visual C# project, you want to leave the Visual C# program and the design environment. It is the same procedure used by nearly all Windows applications:

- Select **File** in the main menu.
- Select **Exit** (at the end of the File menu).

Stop Visual C# now. Visual C# will close all open windows and you will be returned to the Windows desktop. In stopping Visual C# with **Sample** active, you may be asked if you want to save certain files. Answer **No**. Like with stopping a project, an alternate way to stop Visual C# is to click on the close button in the upper right hand corner of the main window. It's the button that looks like an **X**.



We covered a lot of new material here, so if you are, that's OK. As we said earlier, you learned a lot of new words and concepts. Don't worry if you don't remember everything we talked about here. You will see the material many times again. It's important that you just have some concept of what goes into a Visual C# project and how it works. And you know how to start and stop Visual C# itself.

In summary, we saw that a Visual C# project is built upon a **form**. **Controls** (also called **objects**) are placed on the form that allow the user and computer to interact. The user generates **events** with the controls that allow the computer to do its job. In the next class, you will begin to acquire the skills that will allow you to begin building your own Visual C# projects. You will see how the parts of a project fit together. Using project **Sample** as an example, you will learn how to locate important parts of a project. Then, in Class 3, you will actually build your first project!