Q. Can I show pictures of file types other than BMP and JPG?

A. Yes. The PictureBox supports the display of images with the extensions BMP, JPG, ICO, EMF, WMF, and GIF. The PictureBox can even save images to a file using any of the supported file types.

Q. Is it possible to show pictures in other controls?

A. The PictureBox is the control to use when you are just displaying images. However, many other controls allow you to display pictures as part of the control. For instance, you can display an image on a button control by setting the button's Image property to a valid picture.

Q. How can I easily get more information about a property when the Description section of the Properties window just doesn't cut it?

A. Click the property in question to select it, and then press F1context-sensitive help applies to properties in the Properties window, as well.

Q. I find that I need to see a lot of design windows at one time, but I can't find that "magic" layout. Any suggestions?

A. Run at a higher resolution. Personally, I won't develop in less than 1024x768. As a matter of fact, all my development machines have two displays, both running at 1152x864. You'll find that any investment you make in having more screen real estate will pay you big dividends.

Q. Is there an easy way to get help about an object's member?

A. Absolutely. Visual C#'s context-sensitive Help extends to code as well as to visual objects. To get help on a member, write a code statement that includes the member (it doesn't have to be a complete statement), position the cursor within the member text, and press F1. For instance, to get help on the int data type, you could type int, position the cursor within the word int, and press F1.

Q. Are there any other types of object members besides properties and methods?

A. Yes. An event is actually a member of an object, although it's not always thought of that way. Although not all objects support events, most objects do support properties and methods.

Q. Is it possible to create custom events for an object?

A. Yes, you can create custom events for your own objects (you'll learn about such objects in Hour 16, "Designing Objects Using Classes"), and you can also create them for existing objects. Creating custom events, however, is beyond the scope of this book.

Q. Is it possible for objects that don't have an interface to support events?

A. Yes. To use the events of such an object, however, the object variable must be dimensioned a special way or the events aren't available. This gets a little tricky and is beyond the scope of this book.

Q. How many form properties should I define at design time as opposed to runtime?

A. You should set all properties that you can at design time. First, it'll be easier to work with the form because you can see exactly what the user will see.

Also, debugging is easier because there's less code.

Q. Should I let the user minimize and maximize all forms?

A. Probably not. First, there's no point in letting a form be maximized if the form isn't set up to adjust its controls accordingly. About forms, print dialog boxes, and spell check windows are examples of forms that should not be resizable.

Q. Do I need to worry about the anchoring and scrolling capabilities of every form I create?

A. Absolutely not. The majority of forms in most applications are dialog boxes. A dialog box is a modal form used to gather data from the user. A dialog box is usually of a fixed size, which means that its border style is set to a style that can't be sized. With a fixed-size form, you don't need to worry about anchoring or scrolling.

++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

Q. How do I know whether a project is a candidate for an MDI interface?

A. If the program will open many instances of the same type of form, it's a candidate for an MDI interface. For example, if you're creating an image-editing program and the intent is to enable the user to open many images at once, MDI makes sense. Also, if you'll have many forms that will share a common toolbar and menu, you might want to consider MDI.

Q. Can I place radio buttons directly on a form?

A. Yes. The form is a container, so all radio buttons placed on a form are mutually exclusive to one another. If you wanted to add a second set of mutually exclusive buttons, they'd have to be placed on a container control. In general, I think it's best to place radio buttons in a group box rather than on a form because the group box provides a border and a caption for the radio buttons and makes it much easier to move around the set of radio buttons when you're designing the form (you simply move the group box).

Q. I've seen what appear to be list boxes that have a check box next to each item in the list. Is this possible?

A. Yes. In Visual C# 2005, this is accomplished using an entirely different control: the checked list box.

Q. What if I need a lot of timers, but I'm concerned about system resources?

A. When possible, use a single timer for multiple duties. This is easy when two events occur at the same intervalwhy bother creating a second timer? When two events occur at different intervals, you can use some decision skills along with static variables (discussed in Hour 11) to share Timer events.

Q. What else can I do with an Image List control?

A. You can assign a unique picture to a node in a Tree View control when the node is selected. You can also display an image in the tab of a tab page in a Tab control. There are many uses, and as you learn more about advanced controls,

you'll see additional opportunities for using images from an Image List.

Q. I have a number of forms with nearly identical menus. Do I really need to take the time to create menus for all these forms?

A. Not as much as you might think. Create a MenuStrip control that has the common items on it, and then copy and paste the control to other forms. You can then build on this menu structure, saving you a lot of time. Be aware though, that when you copy and paste a control, the corresponding code does not get copied.

Q. I've seen applications that allow the end user to customize the menus and toolbars. Can I do that with the Visual C# menus and toolbars?

A. Yes and no. There is no built-in functionality to do this, so you would have to write LOTS of complicated code to allow users to customize your toolbars. A better approach is to purchase a third-party component that provides this functionality. Personally, I think buying a component that supports this is a much better option than attempting to write the code yourself.

Q. Do I need to pay much attention to scope when defining my procedures?

A. It might be tempting to create all your methods as public, but this is bad coding practice for a number of reasons. For one thing, you'll find that in larger projects, you have methods with the same name that do slightly different things. Usually, these routines are relevant only within a limited scope. However, if you create all public methods, you'll run into conflicts when you create a method with the same name in the same scope. If the method isn't needed at the public level, don't define it for public access.

Q. What is a "reasonable" number of classes?

A. This is hard to say. There really is no right answer. Instead of worrying about an exact count, you should strive to make sure that your classes are logical and that they contain only appropriate methods and properties.

++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

Q. Are any performance tricks related to the many data types?

A. One trick when using whole numbers (values with no decimal places) is to use the data type that matches your processor. For instance, most current home and office computers have 32-bit processors. The Visual C# Integer data type is made up of 32 bits. Believe it or not, Visual C# can process an int variable faster than it can process a short variable, even though the short variable is smaller. This has to do with the architecture of the CPU, memory, and bus. The explanation is complicated, but the end result is that you should usually use int rather than short, even when working with values that don't require the larger size of the int.

Q. Are arrays limited to two dimensions?

A. Although I showed only two dimensions (that is, intMeasurements[3,1]), arrays can have many dimensions, such as intMeasurements[3,3,3,4]. The technical maximum is 60 dimensions, but you probably won't use more than three.

Q. Should I always specify parentheses to ensure that operators are evaluated as I expect them to be?

A. Visual C# never fails to evaluate expressions according to the order of operator precedence, so using parentheses isn't necessary when the order of precedence is correct for an expression. However, using parentheses assures you that the expression is being evaluated the way you want it to, and might make the expression easier to read by other people. This really is your choice.

Q. I would like to learn more about the properties and methods available in the DateTime structure; where can I find all the members listed?

A. I would look at the DateTime members documentation found within the .NET Framework documentation. This is available on the MSDN site and as an installable option when installing Visual C#.

Q. What if I want to execute code only when an expression in an if statement is false, not true? Do I need to place the code in an else clause, and no code after the if?

A. This is where Boolean logic helps. What you need to do is make the expression evaluate to true for the code you want to run. This is accomplished using the not operator (!) in the expression, like this:

```
if (!expression)
. . .
```

Q. How important is the order in which case statements are created?

A. This all depends on the situation. In the earlier example in which the selected animal was considered and the number of legs it has was displayed, the order of the Dog and Horse case was important. If all case statements contained code, the order has no effect.

Q. Is there ever a situation where you would want a loop to run indefinitely?

A. Game programmers often create a single loop that runs indefinitely, and all logic and user input takes place in this main loop. Other than such a very specific situation, all loops should terminate at some point.

Q. Should I be concerned about the performance differences between the two types of loops?

A. With today's fast processors, chances are good that the performance difference between the two loop types in any given situation will be overshadowed by the readability and functionality of the best choice of loop. If you have a situation in which performance is critical, write the loop using all the ways you can think of, benchmark the results, and choose the fastest loop.

Q. Should I alert the user that an exception has occurred or just let the code keep running?

A. If you've written code to handle the specific exception, there's probably no need to tell the user about it. However, if an exception occurs that the code doesn't know how to address, you should provide the user with the exception information so that they can report the problem accurately so that you can fix it.

Q. Should I comment every statement in my application?

A. Probably not. However, consider commenting every decision-making and looping construct in your program. Such sections of code are usually pivotal to the success of the procedure, and what they do isn't always obvious.

++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
Q. Should I always try to place code into instance classes rather than static classes?

A. Not necessarily. As with most things, there are no hard and fast rules. Correctly programming instance classes takes some skill and experience, and programming static is easier for the beginner. If you want to experiment with instance classes, I encourage you to do so. However, don't feel as though you have to place everything into instantiated classes.

Q. I want to create a general class with a lot of miscellaneous methodssort of a "catchall" class. What's the best way to do this?

A. If you want to create some sort of utility class, I recommend calling the class something like clsUtility. Then you can use this class throughout your application to access the utility functions.

Q. Is it possible to capture keystrokes at the form level, rather than capturing them in control events?

A. Yes. For the form's keyboard-related events to fire when a control has the focus, however, you must set the form's KeyPreview property to true. The control's keyboard events will still fire, unless you set KeyPressEventArgs.Handled to true in the control's KeyPress event.

Q. You don't seem to always specify a button in your MessageBox.Show() statements throughout this book. Why?

A. If you don't explicitly designate a button or buttons, Visual C# displays the OK button. Therefore, if all you want is an OK button, you don't need to pass a value to the Buttons argument.

Q. What if I need to draw a lot of lines, one starting where another ends? Do I need to call DrawLine() for each line?

A. The Graphics object has a method called DrawLines(), which accepts a series of points. The method draws lines connecting the sequence of points.

Q. Is there a way to fill a shape?

A. The Graphics object includes methods that draw filled shapes, such as FillEllipse() and FillRectangle().

Q. What if I want to perform an operation on a file, but something is preventing the operation, such as the file might be open or I don't have rights to the file?

A. All the method calls have one or more exceptions that can be thrown in the event that the method fails. These method calls are listed in the online help. You can use the techniques discussed in Hour 15 to trap the exceptions.

Q. What if a user types a filename into one of the file dialog boxes, but the user doesn't include the extension?

A. By default, both file dialog controls have their AddExtension properties set to true. When this property is set to TRue, Visual C# automatically appends the extension of the currently selected filter.

Q. Can I use a text file to save configuration information?

A. Yes, you could do that. You would need some way to denote the data element. For example, how would you know that the first line was the BackColor setting, as opposed to a default file path, for example? One method would be to append the data element to a caption, as in BackColor=White. You would then have to parse the data as you read it from the text file. The Registry is probably a better solution for something like this, but a text file could be useful if you wanted to transfer settings to a different computer.

Q. Can I store binary data instead of text to a file?

A. Visual C# 2005 includes classes designed to work with binary files: BinaryWriter and BinaryReader. You would need to use objects based on these classes, instead of using StreamWriter and StreamReader objects.

++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

Q. If I want to connect to a data source other than Jet, how do I know what connection string to use?

A. Not only is different connection information available for different types of data sources but also for different versions of different data sources. The best way of determining the connection string is to consult the documentation for the data source to which you want to attach.

Q. What if I don't know where the database will be at runtime?

A. For file-based data sources such as Jet, you can add an Open File Dialog control to the form and let the user browse and select the database. Then concatenate the filename with the rest of the connection information (such as the provider string).

Q. What are some applications that support automation?

A. All the Microsoft Office products, as well as Microsoft Visio, support automation. You can create a robust application by building a client that uses multiple automation servers. For example, you could calculate data in Excel and then format and print the data in Word.

Q. Is it possible to create an Automation server so that my application can be controlled by others?

A. Yes, it is possible to create .NET components that can be used by other applications. If this interests you, I suggest you seek out an advanced text on the subject.

Q. How can I create the great installation wizards I see other install applications use?

A. If you want to create robust installations that gather user input in wizards, make changes to the Registry, allow you to include additional files, create shortcuts, and so on, you need to use a tool that uses the Windows Installer technology.

Q. Should I assume that a user will always have the .NET Framework on her computer?

A. Generally, no. When distributing updates to your project, it's probably a safe bet that the user has installed the .NET Framework. For an initial installation, you should specify the .NET Framework as a prerequisite (note that this is set by default).