Syntax Errors

When you break VB's rules for punctuation, format, or spelling, you generate a syntax error. Fortunately, the smart editor finds most syntax errors and even corrects many of them for you. The syntax errors that the editor cannot identify are found and reported by the compiler as it attempts to convert the code into intermediate machine language. A compiler-reported syntax error may be referred to as a *compile error*.

The editor can correct some syntax errors by making assumptions and not even report the error to you. For example, a string of characters must have opening and closing quotes, such as "Hello World". But if you type the opening quote and forget the closing quote, the editor automatically adds the closing quote when you move to the next line. And if you forget the opening and closing parentheses after a method name, such as `Close()`, again the editor will add them for you when you move off the line. Of course, sometimes the editor will make a wrong assumption, but you will be watching, right?

The editor identifies syntax errors as you move off the offending line. A blue squiggly line appears under the part of the line that the editor cannot interpret. You can view the error message by pausing the mouse pointer over the error, which pops up a box that describes the error (Figure 1.49). You also can display an Error List window (*View / Error List*), which appears at the bottom of the Editor window and shows all error messages along with the line number of the statement that caused the error. You can display line numbers on the source code (Figure 1.50) with *Tools / Options*. If *Show All Settings* is selected in the *Options* dialog box, choose *Text Editor / Basic* and check *Line Numbers*; if *Show All Settings* is not checked, then choose *Text Editor / Basic / Editor* and check *Line Numbers*.

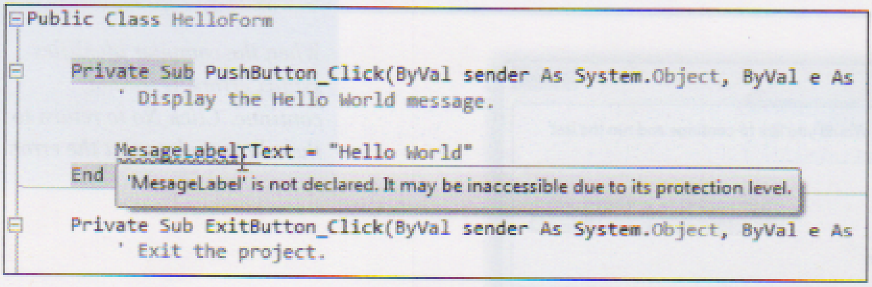
The quickest way to jump to an error line is to point to a message in the Error List window and double-click. The line in error is selected in the Editor window and a list of suggested corrections pops up.

At times the editor can recognize errors and offer suggested solutions. This is more likely to occur in later chapters as you begin to use new keywords. In Chapter 3 you learn to declare elements that can use a data type called *Decimal*. If you accidentally mistype the word Decimal, a small red line appears at the end of the word. Point to the line and an **AutoCorrect** box appears, offering to change the word to the correct spelling. Figure 1.51 shows AutoCorrect in action.

✓ TIP

The Visual Basic AutoCorrect feature can suggest corrections for common syntax errors. ■

If a syntax error is found by the compiler, you will see the dialog box shown in Figure 1.52. Click No and return to the editor, correct your errors, and run the program again.



```
Public Class HelloForm

    Private Sub PushButton_Click(ByVal sender As System.Object, ByVal e As
        ' Display the Hello World message.

        MesageLabel.Text = "Hello World"
    End
```

'MesageLabel' is not declared. It may be inaccessible due to its protection level.

```
    Private Sub ExitButton_Click(ByVal sender As System.Object, ByVal e As
        ' Exit the project.
```

Figure 1.49

The editor identifies a syntax error with a squiggly blue line, and you can point to an error to pop up the error message.