STEP 2 Click the Start Debugging button on the Standard toolbar.

The program is compiled and saved, and then is run on the computer. When the program runs, the Hotel Room Selection window is displayed on the screen (Figure 3-23). Notice that the Standard Room image is not displayed in the window because the Visible property for the picStandardRoom PictureBox object was set to False (see Figure 3-19 on page 126). Notice also that the Select Room button is dimmed, which indicates its Enabled property is set to False.

FIGURE 3-23

ONLINE REINFORCEMENT

To view a video of the process in the previous steps, visit scsite.com/vb2010/ch3 and then select Figure 3-22.

Once you have started running the program, it will continue to run until you close it. To close a program, click the Close button on the right of the window title bar (see Figure 3-23).

Once you have set all the properties for the objects in the user interface, the design of the user interface is complete. You now are ready to move to the next phase of the program development life cycle — designing the program processing objects.

Visual Basic Program Coding

Prior to beginning the design of the program processing objects, the developer must understand certain program coding principles and techniques so he or she can apply this knowledge to the program design. **Program code** is the set of instructions written by the developer that direct the program to carry out the processing required in the program. The following sections explain the Visual Basic 2010 code required for the Hotel Room Selection program.

Entering Visual Basic Code for Event Handling

As you have learned, most program processing in an event-driven program occurs when the user triggers an event. For example, when a user clicks a button on the graphical user interface, this activity can trigger an event and the program performs

the required processing. The developer writes program code to carry out the processing. This code is placed in a section of the program called an **event handler** — so-called because it "handles" the event that the user action triggers by executing code that performs the required processing.

To write the code for an event handler, the developer first must identify the GUI object that will be used to trigger the event. For example, in the sample program, you have learned that when the Standard Room button is clicked, the standard room picture should appear in the picStandardRoom PictureBox object. To write the code that will display the standard room picture, the developer must inform Visual Studio that the Standard Room button is the object for which the code is to be written, and that an event handler must be created for the click event. This can be done using the following steps:

STEP 1 With Visual Studio 2010 and the Hotel Room Selection program open and the frmHotelRoomSelection.vb [Design] tabbed window visible, point to the Standard Room Button object in the Windows Form object.

The pointer points to the Standard Room Button object (Figure 3-24). The four-headed arrow pointer indicates you can drag the Button object to another location in the window if desired.

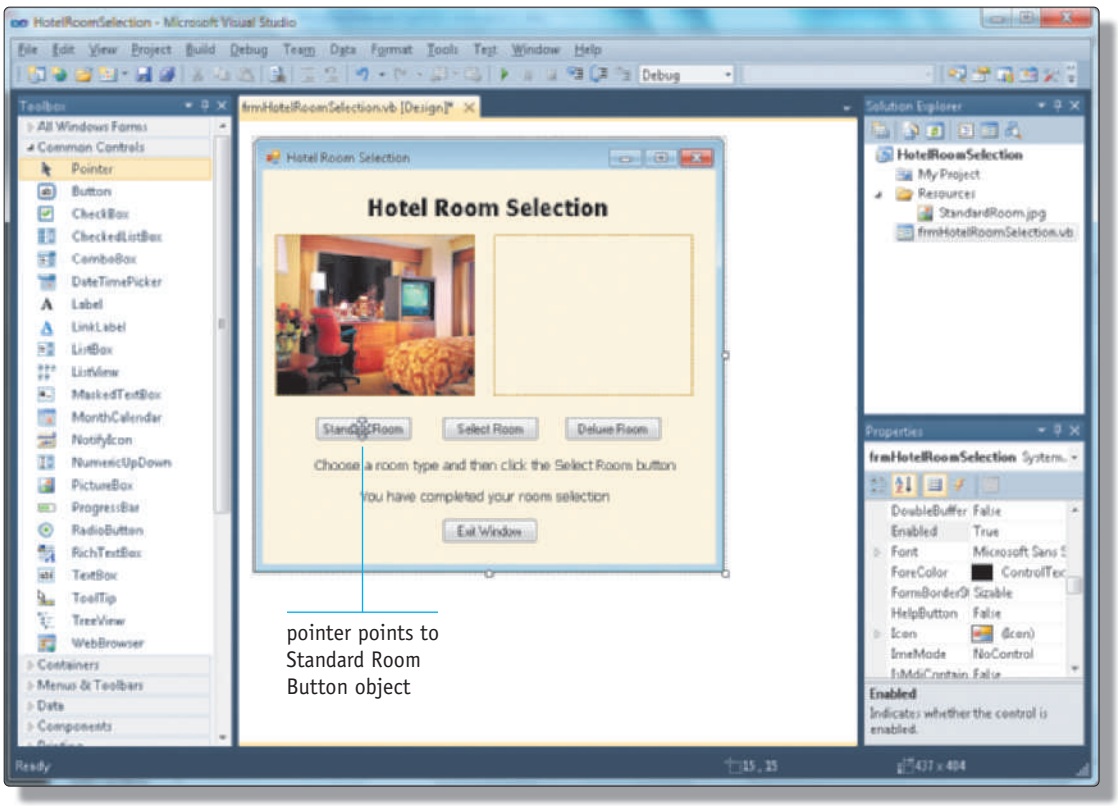

FIGURE 3-24

STEP 2 Double-click the Standard Room Button object.

The code window is displayed on the frmHotelRoomSelection.vb tabbed page (Figure 3-25). The code in the window is generated by Visual Studio. This code identifies an event handler, which is the code that executes when an event is triggered. In this case, when the Standard Room button is clicked, the code in this event handler will be executed by the program. The list box at the upper-left of the tabbed page identifies the object for which the event hander will execute — in this case, the btnStandardRoom object. The list box at the upper-right of the tabbed page identifies the event that must occur in order to execute the code in the event handler. The event identified in Figure 3-25 is Click. So, when the user clicks the Standard Room button, the program will execute the code between the Private Sub statement and the End Sub statement. In Figure 3-25, no code other than the event handler identification code generated by Visual Studio has been entered. The insertion point is located where the developer should begin entering the code that executes when the user clicks the btnStandardRoom Button object.*

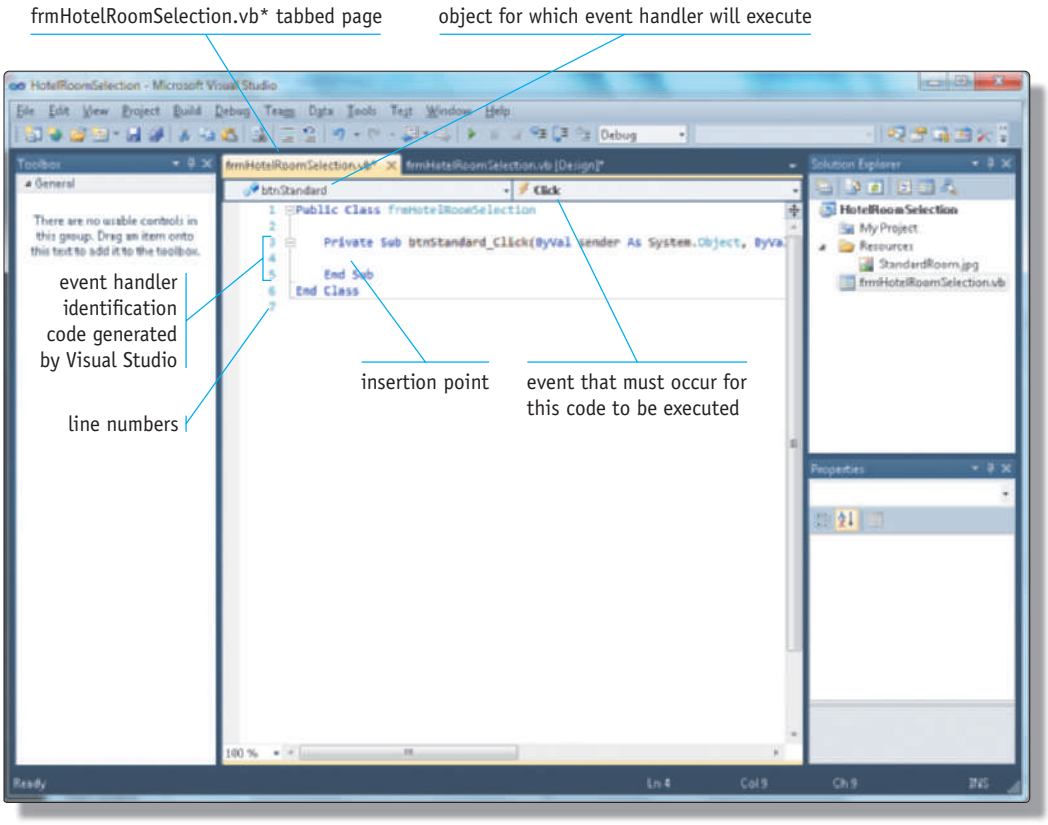

FIGURE 3-25

HEADS UP

In the left column of the coding window in Figure 3-25, line numbers appear. These line numbers help identify each line of code in the coding window. They do not appear by default, however, so you must instruct Visual Studio to display the line numbers. If line numbers do not appear in the coding window on a computer you are using, you can display them by completing the following steps: 1) Click Tools on the menu bar; 2) Click Options on the Tools menu; 3) If necessary, click the triangle next to Text Editor in the Options dialog box; 4) If necessary, click the triangle next to Basic in the list below Text Editor; 5) Click Editor in the list below Basic; 6) Place a check mark in the Line numbers check box; 7) Click the OK button in the Options dialog box.

ONLINE REINFORCEMENT

To view a video of the process in the previous steps, visit scsite.com/vb2010/ch3 and then select Figure 3-24.

Visual Basic 2010 Coding Statements

A Visual Basic 2010 coding statement contains instruction(s) that the computer eventually executes. Visual Basic has a set of rules, or **syntax**, that specifies how each statement must be written.

In the Hotel Room Selection program, you will recall that when the user clicks the Standard Room Button while the program is running, the standard room image should be displayed in the picStandardRoom PictureBox object. Figure 3-26 shows a Visual Basic coding statement that sets the Visible property for the picStandardRoom PictureBox object to True so the image is displayed in the picture box after the statement is executed.

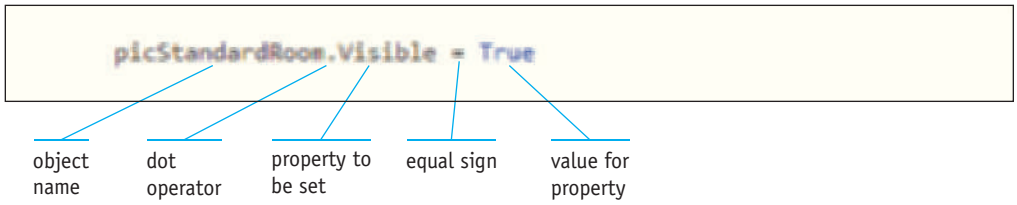

FIGURE 3-26

The first part of the statement, picStandardRoom, identifies the object containing the property to set. The name of the object is followed by the dot operator (period) with no intervening spaces. The dot operator separates the name of the object from the next entry in the statement and is required.

Following the dot operator is the name of the property to set. In Figure 3-26, the name of the property is Visible. You will recall that the Visible property determines whether an image is displayed in the PictureBox object when the program is running. In Figure 3-19 on page 126, the Visible property for the picStandardRoom PictureBox object was set to False so the image would not be displayed when the program was started. This statement sets the Visible property to True so the image will be displayed.

The property name is followed by a space and then an equal sign. The space is not required, but good coding practice dictates that elements within a statement should be separated by a space so the statement is easier to read. One or more spaces can be used, but most developers use only one space. The equal sign is required because it indicates that the value to be used for setting the property follows. A space follows the equal sign for ease of readability.

The word True follows the space. The value True in the Visible property indicates that the image in the PictureBox object should be displayed. When the program is running, as soon as the Visible property is set to True, the image appears in the picture box.

Each entry within the program statement must be correct or the program will not compile. This means the object name must be spelled properly, the dot operator must be placed in the correct location, the name of the property must be spelled properly, the equal sign must be present, and a valid entry for the property must follow the equal sign. For the Visible property, the only valid entries are True and False, so the word True or the word False must follow the equal sign.

General Format of a Visual Basic Statement

The general format of the Visual Basic statement shown in Figure 3-26 appears in Figure 3-27.

General Format: Property Value Assignment Statement

```
objectname.property = propertyvalue
```

EXAMPLE	*RESULT*
picStandardRoom.Visible = True	Picture is visible
btnSelectRoom.Enabled = False	Button is dimmed

FIGURE 3-27

In the general format, the object name always is the first item in the Visual Basic statement. The object name is the name you specified in the (Name) property in the Properties window. In Figure 3-26, the object name is picStandardRoom because that is the name given to the standard room PictureBox object.

The dot operator (period) is required. It follows the object name with no space between them. Immediately following the dot operator is the name of the property that will be set by the statement. The property name must be spelled correctly and must be a valid property for the object named in the statement. Valid properties that can be specified in the statement are identified in the Properties window associated with the object.

The equal sign must follow zero or more spaces in the statement. Visual Basic statements do not require spaces, nor is there is a limit on how many spaces can be contained between elements in the statement. The equal sign identifies the statement as an **assignment statement**, which means the value on the right side of the equal sign is assigned to the element on the left side of the equal sign. When setting properties, it means the value on the right side of the equal sign is assigned to the property identified on the left side of the equal sign.

The property value specified in the assignment statement must be a valid value for the property identified on the left side of the equal sign. You can see the valid values for a given property by looking in the Properties window for the object whose property you are setting.

After you have entered the property value, the Visual Basic statement is complete. Because correct programming protocol states that only one statement should appear on a line, the next step is to press the ENTER key to move the insertion point to the next line in the coding window.

The general statement format shown in Figure 3-27 is used for all statements in which the code sets the value of an object property.

HEADS UP

Unlike some programming languages, in Visual Basic capitalization of object names, properties, property values, and other values is not important. Therefore, the object name picStandardRoom is equivalent to the object name picstandardroom. The reason for capitalization within the object name is to make the name easier to read.

IN THE REAL WORLD

Microsoft created IntelliSense in response to the needs of developers in a rapid application development environment so they can enter statements accurately and easily. Most developers use IntelliSense and its use is the standard within the software industry.

IntelliSense

In Figure 3-25 on page 131, the insertion point is located in the coding window. To enter the statement in Figure 3-26 into the actual program in the coding window, you can type the entire statement. Visual Studio, however, provides help when entering a statement so that you will be less prone to make an error when entering the statement. This help is called IntelliSense.

IntelliSense displays all allowable entries you can make in a Visual Basic statement each time a dot (period), equal sign, or other special character required for the statement is typed. When you type the prefix pic as shown in Figure 3-28, an IntelliSense window opens with all the objects that begin with that prefix. Instead of possibly misspelling the object name, you can select it from the IntelliSense list. Therefore, when using IntelliSense, the complete Visual Basic statement would be as shown in Figure 3-28:

```
picStandardRoom.Visible = True
```

FIGURE 3-28

When you type the first few letters of the object name, IntelliSense displays a list of all the entries, including all the objects, that can be specified in the statement.

Enter a Visual Basic Statement

To enter the Visual Basic statement in Figure 3-28 using IntelliSense, you can complete the following steps:

STEP 1 With the code editing window open and the insertion point positioned as shown in Figure 3-25 on page 131, type `pic.`

The characters pic are displayed in the code window (Figure 3-29). IntelliSense displays a list of all the entries that can follow the prefix in the statement. Sometimes the entry selected in the list is the correct entry for the statement you are entering, but often it is not the correct entry. Therefore, you must identify the correct statement in the list before entering it.

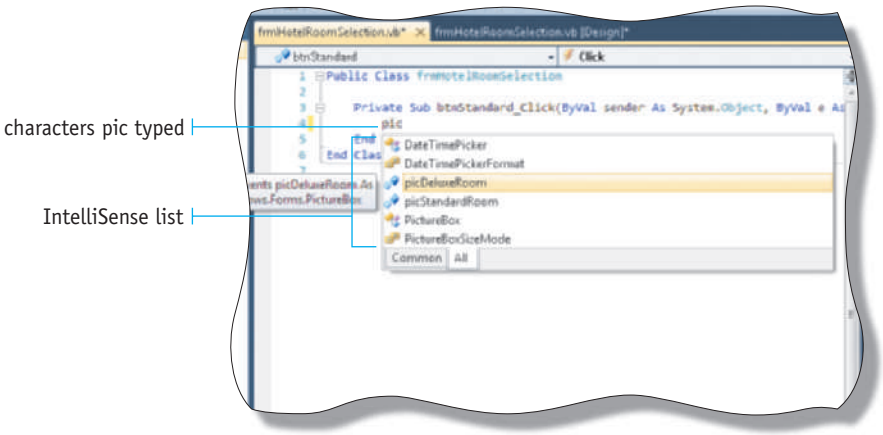

FIGURE 3-29

STEP 2 To identify the correct entry, type the next letter of the entry until the entry is selected. In this case, type `S` on your keyboard.

When you type characters, IntelliSense highlights in the list an entry that begins with the letters you type (Figure 3-30). When you enter pics, IntelliSense highlights the only term in the list that begins with pics, which is picStandardRoom. This is the object name you want to enter into the Visual Basic statement.

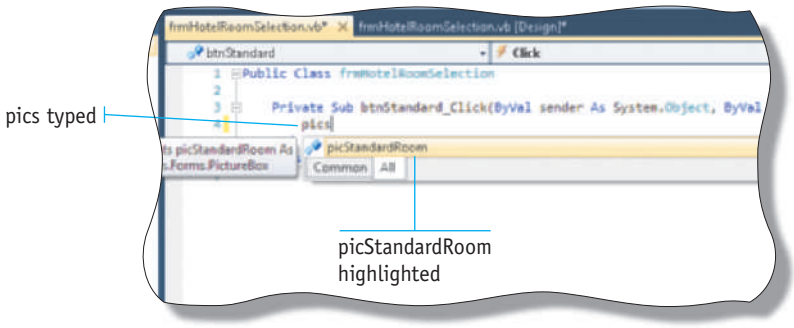

FIGURE 3-30

STEP 3 When IntelliSense highlights the correct object name, press the key on the keyboard corresponding to the entry that is to follow the object name. In this case, press the PERIOD key.

IntelliSense automatically enters the entire object name into the Visual Basic statement and the period (the character you typed) following the object name (Figure 3-31). In addition, because IntelliSense realizes that the dot you entered means more information is required in the statement, a list of the allowable entries following the dot is displayed.

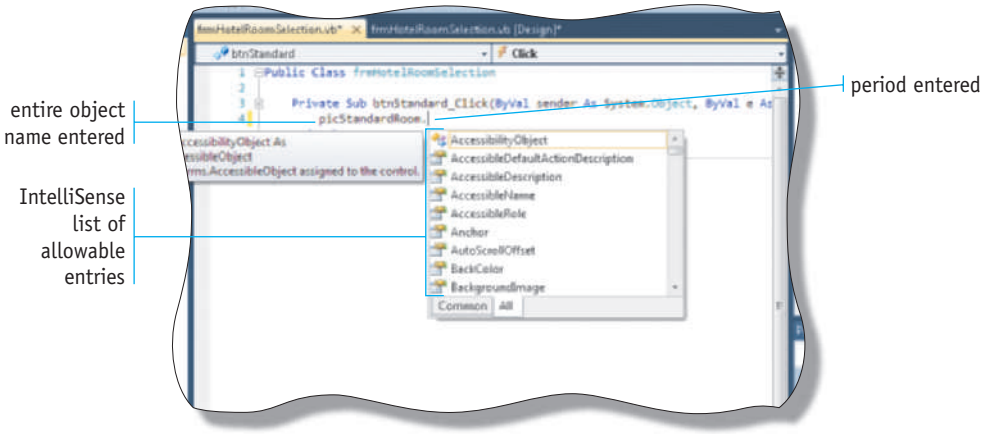

FIGURE 3-31

STEP 4 As with the object name in Step 2, the next step is to enter one or more characters until IntelliSense highlights the desired property in the list. Type the letter v on your keyboard.

IntelliSense highlights the properties in the list that begins with the letter v (Visible), or that contains the letter v such as in ProductVersion (Figure 3-32). Because the Visible property is highlighted, no further action is required to select the Visible property.

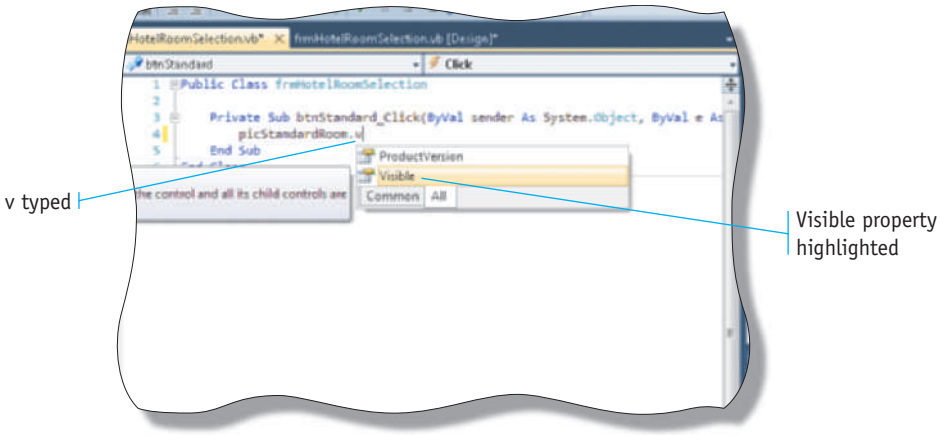

FIGURE 3-32

IN THE REAL WORLD

When you enter a statement using IntelliSense, by default IntelliSense will format the statement after you press the ENTER key. So, if you did not enter spaces in the statement before and after the equal sign, IntelliSense automatically would insert the spaces when you press the ENTER key. You can choose whether to enter spaces as you enter the statement, or let IntelliSense insert the spaces when you press the ENTER key.

STEP 5 Press the key for the character that is to follow the property name. In this case, press the SPACEBAR on the keyboard.

IntelliSense enters the highlighted property name (Visible) followed by the character you typed (space) (Figure 3-33). The space indicates to Visual Basic that the object name and property name entry is complete. Notice also that the IntelliSense tip specifies what the statement will be able to do. In Figure 3-33, it states that the statement "gets or sets a value indicating whether the control is displayed." This means the Visible property indicates whether the picStandardRoom PictureBox object is displayed.

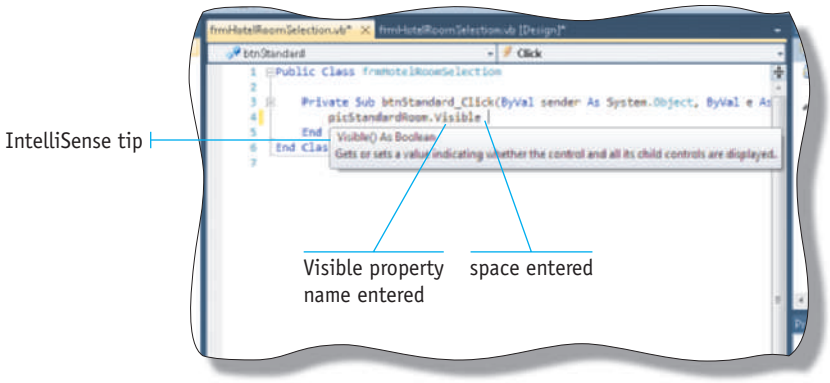

FIGURE 3-33

STEP 6 Press the EQUAL SIGN key on the keyboard and then press the SPACEBAR. On the IntelliSense list, click the Common tab to display the most common results.

The equal sign and a space are displayed and then IntelliSense displays a list containing the entries you can make (Figure 3-34). For the Visible property, the only possible entries following the equal sign are False (which indicates the PictureBox object should not be visible) and True (which indicates the PictureBox object should be visible).

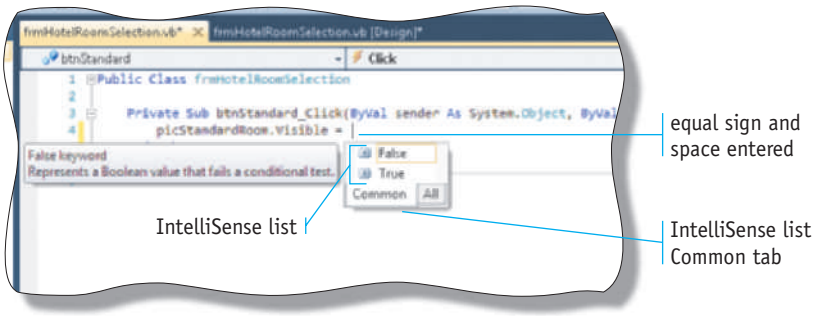

FIGURE 3-34

STEP 7 Type `t` on the keyboard.

IntelliSense highlights the True entry (Figure 3-35).

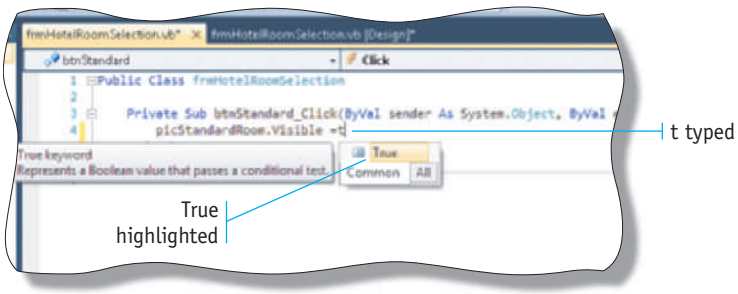

FIGURE 3-35

STEP 8 Press the key for the character that is to follow the True entry. In this case, press the ENTER key.

Because you pressed the ENTER key, IntelliSense enters True into the statement and then Visual Studio moves the indented insertion point to the next line (Figure 3-36). The Visual Basic statement now is completely entered.

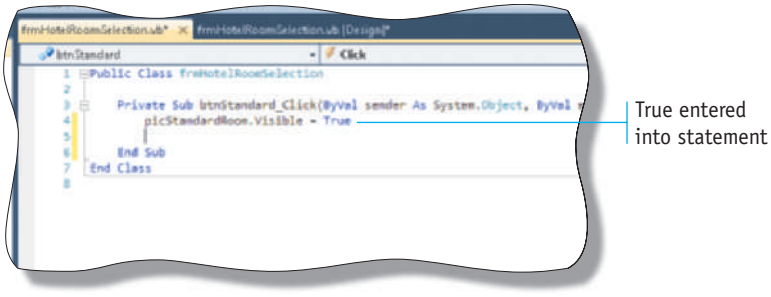

FIGURE 3-36

IN THE REAL WORLD

You can highlight the entries in the lists IntelliSense displays in a variety of ways. In the example, typing the first character of the entry was used. Other ways to highlight an entry include pressing the UP ARROW key and the DOWN ARROW key until the entry is highlighted, and using the mouse to scroll in the list until the entry is visible and then clicking the entry.

To enter an entry from the list, the example illustrated pressing the key for the character to follow the entry from the list. Other ways include pressing the Tab key, and double-clicking the entry in the list. In both these cases, you still must enter the next character in the statement, whereas in the example, that character is entered when you press the correct key to select the IntelliSense entry from the list.

ONLINE REINFORCEMENT

To view a video of the process in the previous steps, visit scsite.com/vb2010/ch3 and then select Figure 3-29.

Visual Studio and IntelliSense automatically create the indentations in the program statements in Figure 3-36 because the indentations make the statements easier to read and understand. As programs become more complex, proper indentation of program statements can be an important factor in developing error-free programs.

The following steps summarize the procedure for using IntelliSense to enter a Visual Basic statement that sets a property:

1. Type the first letter(s) of the name of the object whose property will be set until the object name is selected in the IntelliSense list.
2. Press the PERIOD key.
3. Type the first letter(s) of the name of the property to be set until the name is highlighted in the IntelliSense list.
4. Press the SPACEBAR to complete the first portion of the statement.
5. Press the EQUAL SIGN key.
6. Press the SPACEBAR.
7. Press the first letter(s) of the entry you want to select in the list until the entry is highlighted; or if IntelliSense does not display a list, type the value for the property.
8. Press the ENTER key.

Using IntelliSense to enter a Visual Basic statement provides two significant advantages: 1) It is faster to enter a statement using IntelliSense than it is to enter a statement by typing it; 2) Using IntelliSense reduces the number of errors committed when entering a statement to almost zero. By using only the entries contained on the IntelliSense lists, the developer seldom will make a mistake by entering an invalid entry. In addition, because the entry is chosen from a list, it is not possible for the entry to be misspelled or mistyped.

Entering a programming statement is a fundamental skill of a Visual Basic programmer. You should understand thoroughly how to enter a programming statement using IntelliSense.

Set Visible Property to False

In Figure 3-36 on page 137, the programming statement set the Visible property for the picStandardRoom PictureBox object to True, which will cause the image in the picture box to be displayed when the statement is executed. The statement will be executed when the user clicks the Standard Room button because the statement is within the btnStandardRoom_Click event handler.

Another setting that must take place when the user clicks the Standard Room button is to set the Visible property for the picDeluxeRoom PictureBox to False so the deluxe room picture is not displayed when the standard room picture is displayed. To set the Visible property for the picDeluxeRoom PictureBox object to False, you could complete the steps on the following pages: