


## The “House” Program

The “House” program uses labels to create “Hot Spots”. Labels cover various parts of the picture. When the user moves the mouse to one of the labels, the mouse pointer changes to a hand, and the tool tip pops up with a description of the part. This program requires a picture of a house with a tree. You can use the one provided with this lesson, or find one you like on the Internet:

- Go to [www.google.com](http://www.google.com).
  - Click on Images.
  - Type house and press Enter.
  - Double click a picture you like to open it full size.
  - Right-click and save to disk.
1. Start a new windows application and name it house.
  2. In the property window select any image as the BackgroundImage. Notice that the image is tiled. We would like to use a picture without tiling. Because the user can resize the form, there is no way to keep the image from tiling. Instead of putting the image on the form we will use a panel.
  3. In the property window, delete the image by selecting the property and pressing delete.


### Panels

A panel can serve as a container for other controls.

1. Add a panel  to the form.
2. In the property window select the BackgroundImage for Panel1 (Panel1.BackgroundImage)
3. Double click the three dots (ellipsis) and select one of the icons in .../graphics/Icons/\*.ico  
The background image *tiles* the panel.
4. Click the ellipsis again and select the picture of a house that you have saved.  
Adjust the size of the panel so that the picture fits perfectly.
5. We will add a label, but we want it to be inside the panel. Instead of double clicking on the label icon **A** in the toolbox, click it once and draw the label by dragging the mouse from the top left corner of the tree down to the lower right corner of the tree. The label should more or less covers the tree.
6. In the property window change the following properties of the label:  
Name: lblTree  
BackColor: Select transparent on the Web tab.  
Text: Delete the text  
*Notice that there is no property called ToolTip.*

### ToolTip

The tool tip is the little words in a yellow box that pops up when you move the mouse over an area.

1. In the toolbox, double click on the ToolTip icon . The tool tip does not appear on the form.  
Instead it is in the panel at the bottom of the screen.
2. Select lblTree and scroll down to the ToolTip property and you will see that there is now a property called ToolTip on ToolTip1. Set this property to Tree.

Run the program and move the mouse over the tree. You should see the word “tree” pop up. You may have to be patient: the default properties for the tooltip make it rather slow.

3. Select ToolTip1 and look at the properties. Change ShowAlways to True. Adjust any of the variables that refer to Delay and run the program again. Adjust the values so that you know what they do and find settings you like.

**Experiment:** Add a few more labels for door, window, etc. You only need one tooltip in the whole project unless you want different settings on the delay for the tooltip.

1. Add a second tooltip. Leave the default settings for tooltip2.
2. Change the properties for lblTree so that ToolTip for Tooltip1 is "Tree", and "20' maple" for the second one.

When you run the program you should see "Tree" the first time, and "20' maple" pop up after.

3. Copy the label and name it lblDoor. Position lblDoor so that it is over the door. Make the tooltips “door” and “security lock”.
4. Add a label and set the tooltip for the window.



## Radio Buttons ☺

We could make a fairly useful program just using transparent labels and tool tips. We will modify the program so that you can use radio buttons ☺ to select English, Spanish or German for the tooltips.

1. We will not use the second tooltip for this project. You can delete it.
2. Make the form slightly larger and add 3 radio buttons. Name them radEnglish, radSpanish, radGerman. Set the text to “&English”, “&Spanish”, “&German”.
3. Make the text for the form “A House”.
4. Write the code for radGerman\_click as shown below:

```
Private Sub radGerman_Click(ByVal sender As Object, ByVal e As System.EventArgs) _
    Handles radGerman.Click
    Me.Text = "Ein haus"
    radEnglish.Text = "Englisch"
    radSpanish.Text = "Spanisch"
    radGerman.Text = "Deutsch"
    Me.ToolTip1.SetToolTip(Me.lblDoor, "Tür") 'ü=alt+0252
    Me.ToolTip1.SetToolTip(Me.lblWindow, "Fenster")
    Me.ToolTip1.SetToolTip(Me.lblTree, "Baum")
End Sub 'radGerman_Click
```

Experiment: Run the program, then try writing the rest of the code before you turn the page.

### Special Characters

This program uses special characters such as ü and é. To insert one of these special characters, first find its ANSI value: select **H**elp, **S**earch For Help On..., search for the word ANSI, then select ANSI Character Set. Click on the words "Characters 128-255". Scroll through the list and note the number to the left of the character you need: the letter ü is 252. To insert ü in the code hold down **ALT** and type **0252** on the **numeric** keypad. (*The codes will only work on the numeric keyboard, but will work almost anywhere in Windows, not just Visual Basic! See the table of ANSI codes.*)

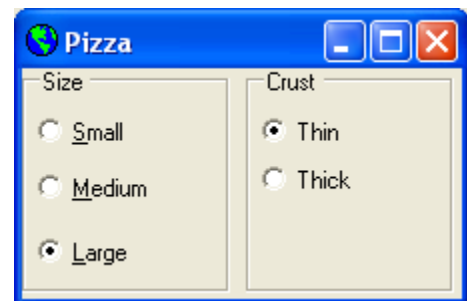
```
Private Sub radSpanish_Click(ByVal sender As Object, ByVal e As System.EventArgs) _
    Handles radSpanish.Click
    Me.Text = "Una casa"
    radEnglish.Text = "inglés" 'é= ALT+0233
    radSpanish.Text = "español" 'ñ= ALT+0241
    radGerman.Text = "alemán" 'á= ALT+0225
    Me.ToolTip1.SetToolTip(Me.lblDoor, "puerta")
    Me.ToolTip1.SetToolTip(Me.lblWindow, "ventana")
    Me.ToolTip1.SetToolTip(Me.lblTree, "arbol")
End Sub 'radSpanish_Click

Private Sub radEnglish_Click(ByVal sender As Object, ByVal e As System.EventArgs) _
    Handles radEnglish.Click
    Me.Text = "A house"
    radEnglish.Text = "English"
    radSpanish.Text = "Spanish"
    radGerman.Text = "German"
    Me.ToolTip1.SetToolTip(Me.lblDoor, "Door")
    Me.ToolTip1.SetToolTip(Me.lblWindow, "Window")
    Me.ToolTip1.SetToolTip(Me.lblTree, "Tree")
End Sub 'radEnglish_Click
```

Save the House program and close it.

### Check Boxes and Radio Buttons

Check boxes are used when the user can make several choices. Radio buttons are used when the user must select only one from a group of choices. Only one radio button in a group can be selected. In the illustration, the radio buttons are placed inside a GroupBox  so that the user can select size from one group and the type of crust from another group.



1. Start a new windows application and name it Pizza.
2. Build the form as shown making sure that each radio button is drawn inside a GroupBox .
3. Name the controls grpSize, containing radSmall, radMedium, and radLarge, and grpCrust containing radThin and radThick. Set the checked property to True for Large and Thin.

- Run the program to make sure you can select one item from each group. If you can only select one button, then delete them and make sure that you draw the buttons inside the GroupBox.

We will use checkboxes for the toppings because you can have as many toppings as you want.

- Add checkboxes named chkPepperoni, chkMushroom, and chkAnchovy, and a label called lblCost as shown below:



### Calculating the Cost

When any change is made, the cost of the pizza will be calculated as follows: a small pizza is \$8, medium is \$10 and large is \$12. Thick crust is \$0.50 additional and each topping is \$1

Since the cost is calculated the same way for any button that is clicked, we will create one procedure to **handle** all of the click events.

- Double click on radMedium to open the code window.
- Change the name from radSmall\_Click to AnyChange.
- Add each of the events to the handles clause: The changes are shown in bold.

```
Private Sub AnyChange(ByVal sender As Object, ByVal e As System.EventArgs) _
    Handles radMedium.Click, radSmall.Click, radLarge.Click, radThick.Click, _
    radThin.Click, chkAnchovy.Click, chkMushroom.Click, chkPepperoni.Click
```

Radio buttons and CheckBoxes have a property Checked that is True when it is selected and false if not. The expression `Me.radSmall.Checked` is a Boolean expression: it does not need to be compared to True to test it.

Although we could write the statement:

```
If Me.radSmall.Checked = True Then Cost = 8
```

The statement can also be written as shown below and this is the preferred method:

```
If Me.radSmall.Checked Then Cost = 8
```

```
Private Sub AnyChange(ByVal sender As Object, ByVal e As System.EventArgs) _
    Handles radMedium.Click, radSmall.Click, radLarge.Click, radThick.Click, _
    radThin.Click, chkAnchovy.Click, chkMushroom.Click, chkPepperoni.Click
    Dim Cost As Double
    If Me.radSmall.Checked Then Cost = 8
    If Me.radMedium.Checked Then Cost = 10
    If Me.radLarge.Checked Then Cost = 12
    If Me.radThick.Checked Then Cost = Cost + 0.5
    If Me.chkAnchovy.Checked Then Cost = Cost + 1
```

```

    If Me.chkMushroom.Checked Then Cost = Cost + 1
    If Me.chkPepperoni.Checked Then Cost = Cost + 1
    Me.lblCost.Text = Cost
End Sub 'AnyChange

```

Run the program and save.

### General Procedures

We have one procedure to handle all changes. If we would like to do something slightly different for each event we could write separate procedures for each event, but we would not want to repeat the code to calculate the cost. One important reason to not repeat code is because it can lead to errors when the code needs to be updated. We will write a general procedure to calculate cost, then call it from the other events.

Since a general procedure is not invoked by an event, type the first line under the end sub for another procedure:

```
Private Sub Calculate()
```

When we hit enter the End Sub line for Calculate is provided for us.

Move the code that calculates the price into calculate, replace it with a call to calculate:

```

Private Sub AnyChange(ByVal sender As Object, ByVal e As System.EventArgs) _
    Handles radMedium.Click, radSmall.Click, radLarge.Click, radThick.Click, _
    radThin.Click, chkMushroom.Click, chkPepperoni.Click
    Calculate()
End Sub 'AnyChange

Private Sub Calculate()
    Dim Cost As Double
    If Me.radSmall.Checked Then Cost = 8
    If Me.radMedium.Checked Then Cost = 10
    If Me.radLarge.Checked Then Cost = 12
    If Me.radThick.Checked Then Cost = Cost + 0.5
    If Me.chkAnchovy.Checked Then Cost = Cost + 1
    If Me.chkMushroom.Checked Then Cost = Cost + 1
    If Me.chkPepperoni.Checked Then Cost = Cost + 1
    Me.lblCost.Text = Format(Cost, "$0.00")
End Sub 'Calculate

```

Remove chkAnchovy\_Click from AnyChange and add the code below:

```

Private Sub chkAnchovy_CheckedChanged(ByVal sender As System.Object, _
    ByVal e As System.EventArgs) Handles chkAnchovy.CheckedChanged
    Me.Text = "Yuck, anchovies!"
    Calculate()
End Sub 'chkAnchovy_CheckedChanged

```

## Formatting Numbers



If you tested the program thoroughly, you should have noticed that when we select a thick crust the cost is displayed with just a 5 after the decimal place. We will solve this problem by specifying the cost be displayed with a \$ and 2 decimal places.

1. Change the last statement in the procedure to:  
`Me.lblCost.Text = Format(Cost, "$0.00")`
2. Run the program and try several selections including thick crust to make sure that the program works.
3. Double click on the word "Format" to select it, then press F1 for help. Look at the examples and explanation. Several examples refer to formatting the date.

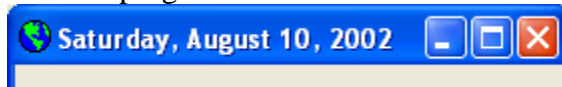
**Experiment:** Add another size, Extra Large, and add a few more toppings.

## Formatting Dates

1. Save the Pizza program, then close it and start a new Windows application called Today.
2. Double click the form and add the line shown below to the form load event:

```
Private Sub Form1_Load(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles MyBase.Load
    Me.Text = Format(Now(), "D")
End Sub 'Form1_Load
```

3. Run the program to see the date as shown below:



**Experiment:** Try each of the formats shown below:

```
Me.Text = Format(Now(), "m") 'August 10
Me.Text = Format(Now(), "Long Time") '11:23:16 AM
Me.Text = Format(Now(), "ddd") 'Sat
Me.Text = Format(Now(), "hh:mm") '11:34
```

You can also format a date other than today. What day were you born? Declare your birthday as shown below:

```
Private Sub Form1_Load(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles MyBase.Load
    Dim MyBirthday As Date = #5/21/1976#
    Me.Text = Format(MyBirthday, "long date")
End Sub 'Form1_Load
```

