CHAPTER 11
EXTRA RESOURCES

Additional Resources

1. "Small Basic Reference Documentation: GraphicsWindow Object" (*http://tiny.cc/graphicswindow/*): Learn about all the properties, events, and methods of the `GraphicsWindow` object.
2. "Small Basic Reference Documentation: Shapes Object" (*http://tiny.cc/shapesobject/*): Review the `Shapes` object to become familiar with its methods!
3. "Special Keys: Alt & F10" (*http://tiny.cc/altf10/*): Learn how ALT and F10 are treated differently in Windows.
4. "More About MouseMove" (*http://tiny.cc/mousemove/*): Dig deeper to discover how the `MouseMove` event works.
5. "Turtle Game Updates" (*http://tiny.cc/turtlegame/*): Share your updates to the Gold Rush turtle game!

Review Questions

1. What is an event?
2. What is the difference between procedural programming and event-driven programming? What are examples of each?
3. What are the six events that the `GraphicsWindow` can check for?
4. What is an event handler?
5. How do you register an event handler?
6. What does Small Basic save in the `LastKey` property when the user presses the 4 key?
7. What does the following code do?

```
Timer.Interval = 2000
```

8. What should you look out for when you're using the `MouseDown` event?
9. BONUS: What is a typewriter?

Practice Exercises

1. What does the following program do? Write the program and run it. How does the `Animate()` method work?

```
GraphicsWindow.MouseDown = OnMouseDown
circle = Shapes.AddEllipse(50, 50)
Sub OnMouseDown
  x = GraphicsWindow.MouseX
  y = GraphicsWindow.MouseY
  Shapes.Animate(circle, x, y, 500)
EndSub
```

2. Run this program:

```
GraphicsWindow.MouseDown = OnMouseDown
GraphicsWindow.MouseUp   = OnMouseUp

Sub OnMouseDown
  Sound.PlayClickAndWait()
EndSub

Sub OnMouseUp
  Sound.PlayBellRingAndWait()
EndSub
```

   a. Click and release the mouse over the graphics window and describe what happens.

b. Click and hold the mouse over the graphics window, and then move the cursor outside the graphics window before releasing the button. Describe what happens.

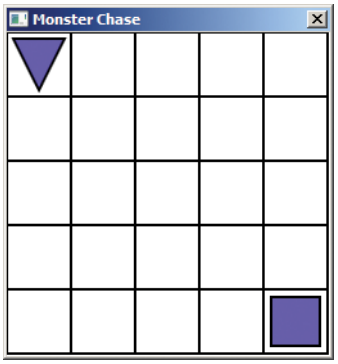
c. What does the outcome of parts a and b tell you about the `MouseDown` and `MouseUp` events?

3. The following simple program tests the typing speed of a user. The program displays ten random letters in the graphics window (one at a time) and it times how long the user takes to click the matching letter on the keyboard. Run the program several times, and then read the source code to understand how it works. Think of some ways to improve it, and try to implement them.

```
' Speed.sb
' Tests your typing speed

GraphicsWindow.Title = "Speed Test"
GraphicsWindow.Width = 300
GraphicsWindow.Height = 300
GraphicsWindow.CanResize = "False"

' Creates a text shape to show the random letters
GraphicsWindow.FontSize = 80
txtID = Shapes.AddText("")
Shapes.Move(txtID, 120, 100)

' Creates a text shape to show the score (total time)
GraphicsWindow.FontSize = 16
scoreID = Shapes.AddText("Total time: ")
Shapes.Move(scoreID, 5, 270)

GraphicsWindow.ShowMessage("Click to Start", "Start")

GraphicsWindow.KeyDown = onKeyDown

t1 = Clock.ElapsedMilliseconds     ' Start time
charNum = 1                        ' Shows only 10 characters

ShowLetter()

Sub ShowLetter
  code = 64 + Math.GetRandomNumber(26)
  char = Text.GetCharacter(code)
  Shapes.SetText(txtID, char)
EndSub

Sub onKeyDown
  If (charNum <= 10) Then
    If (GraphicsWindow.LastKey = char) Then ' Matches
      Sound.PlayClick()
      If (charNum = 10) Then
        ShowElapsedTime()
```

```
      Else
        ShowLetter()
      EndIf
      charNum = charNum + 1
    EndIf
  EndIf
EndSub

Sub ShowElapsedTime
  totTime = Clock.ElapsedMilliseconds - t1
  totTime = totTime / 1000
  msg = "Total time: " + totTime + " sec."
  Shapes.SetText(scoreID, msg)
EndSub
```

4. Write a game in which the player is locked in a cage with a monster (see the following figure).



The player is represented by the square shape and the monster by the triangle. The player uses the arrow keys to move. After each move, the monster moves toward the player. The longer the player can outwit the monster, the higher the player's score!

a. Open the file *MonsterChase_Incomplete.sb* from this chapter's folder. You should see the following code. You'll add the missing sections.

```
' MonsterChase_Incomplete.sb
' Try to escape from a monster

GraphicsWindow.Title     = "Monster Chase"
GraphicsWindow.Width     = 250
GraphicsWindow.Height    = 250
GraphicsWindow.CanResize = 0

Grid()                                      ' Draws a 5x5 grid
P = Shapes.AddRectangle(40, 40)             ' Player's shape
M = Shapes.AddTriangle(0, 0, 40, 0, 20, 40)  ' Monster's shape

GraphicsWindow.KeyDown = OnKeyDown
```

```
NewGame()

Sub NewGame
    ' [TO DO]
EndSub

Sub OnKeyDown
    ' [TO DO]
EndSub

Sub Grid
  For I = 1 To 5
    For J = 1 To 5
      XO = (J - 1) * 50
      YO = (I - 1) * 50
      GraphicsWindow.DrawRectangle(XO, YO, 50, 50)
    EndFor
  EndFor
EndSub
```

b. Add the following NewGame() subroutine where the first [TO DO] comment is. Then add the Move() subroutine after the NewGame() subroutine.

```
Sub NewGame
  rp = 5      ' Player's row
  cp = 5      ' Player's column
  rm = 1      ' Monster's row
  cm = 1      ' Monster's column
  count = 0   ' Number of player's movements

  Move()      ' Positions player and monster on the grid
EndSub
Sub Move
  Shapes.Move(P, (cp - 1) * 50 + 5, (rp - 1) * 50 + 5)
  Shapes.Move(M, (cm - 1) * 50 + 5, (rm - 1) * 50 + 5)
EndSub
```

When you run the game now, you'll see the monster's triangle in the upper-left corner and the player's square in the lower-right corner.

c. Add the following OnKeyDown() subroutine where the second [TO DO] comment is. How is the program stopping the player and monster from moving out of the grid?

```
' Checks whether the player's move is within the grid
' If the player's move is valid, it processes the move
Sub OnKeyDown
  newR = rp                          ' New row
  newC = cp                          ' New column
  key = GraphicsWindow.LastKey
  If (key = "Up" And newR > 1) Then
    newR = newR - 1
  ElseIf (key = "Down" And newR < 5) Then
```

```
    newR = newR + 1
  ElseIf (key = "Left" And newC > 1) Then
    newC = newC - 1
  ElseIf (key = "Right" and newC < 5) Then
    newC = newC + 1
  EndIf

  If (newR <> rp Or newC <> cp) Then  ' Player's position changed
    count = count + 1                 ' Increases move count
    rp = newR                         ' Sets new values
    cp = newC
    Process()
  EndIf
EndSub
```

d. Add the following `Process()` subroutine to the end of the program. How is the program telling the user when the game ends? What does the program do when the game ends?

```
' Processes player's move
' Moves player and then moves monster
Sub Process
  done = 0            ' Assumes the player is not caught by the monster
  Move()              ' Updates the positions in the GUI
  If (rp = rm And cp = cm) Then  ' Player moves to the monster's grid
    done = 1          ' Game over
  Else                ' Player moves and avoids monster
    MoveMonster()     ' Determines monster's new position (row and column)
    Move()            ' Updates positions in GUI
    If (rp = rm And cp = cm) Then ' If monster hits player
      done = 1        ' Game over
    EndIf
  EndIf

  If (done = 1) Then
    GraphicsWindow.ShowMessage("Game over: " + count, "Game over")
    NewGame()
  EndIf
EndSub
```

e. Add the following `MoveMonster()` subroutine to the end of the program. Explain how the monster decides where to move.

```
Sub MoveMonster
  If (rm = rp And cm < cp) Then      ' Player is east of monster
    D = 1                            ' Direction = East
  ElseIf (rm > rp And cm < cp) Then ' Player is northeast of monster
    D = 2                            ' Direction = Northeast
  ElseIf (rm > rp And cm = cp) Then ' Player is directly north of monster
    D = 3                            ' Direction = North
  ElseIf (rm > rp And cm > cp) Then ' Player is northwest of monster
    D = 4                            ' Direction = Northwest
  ElseIf (rm = rp And cm > cp) Then ' Player is west of the monster
```

```
    D = 5                           ' Direction = West
  ElseIf (rm < rp And cm > cp) Then ' Player is southwest of monster
    D = 6                           ' Direction = Southwest
  ElseIf (rm > rp And cm = cp) Then ' Player is south of monster
    D = 7                           ' Dircetion = South
  Else                              ' Player is southeast of monster
    D = 8                           ' Direction = Southeast
  EndIf

  RandDir:
  D = D + (Math.GetRandomNumber(3) - 2) ' Adds 1, 0, or -1 to D
  If (D < 1) Then       ' Makes sure the direction is still between 1 and 8
    D = 1
  ElseIf (D > 8) Then
    D = 8
  EndIf

  ' Moves
  newR = rm
  newC = cm

  If ((D = 2 Or D = 3 Or D = 4) And (newR > 1)) Then     ' Moves up
    newR = newR - 1
  ElseIf ((D = 6 Or D = 7 Or D = 8) And (newR < 5)) Then ' Moves down
    newR = newR + 1
  EndIf

  If ((D = 4 Or D = 5 Or D = 6) And (newC > 1)) Then     ' Moves left
    newC = newC - 1
  ElseIf ((D = 1 or D = 2 Or D = 8) And (newC < 5)) Then ' Moves right
    newC = newC + 1
  EndIf

  If (newR = rm And newC = cm) Then ' Monster doesn't move
    Goto RandDir
  EndIf

  rm = newR ' Set monster's new position (row and column)
  cm = newC
EndSub
```

5. Write a program that lets the user draw in the graphics window using the arrow keys. The cursor should start in the middle of the graphics window. When the user presses an arrow key, the cursor should move in that direction and leave a trail. Also, let the user change the pen's color by pressing the spacebar. (Hint: use the `Turtle` object.)