

Animation (with an Introduction to Indexing)



To make something appear to move on the form, erase it and redraw it a small distance away.

© University of Washington, 2001



A Simple Idea

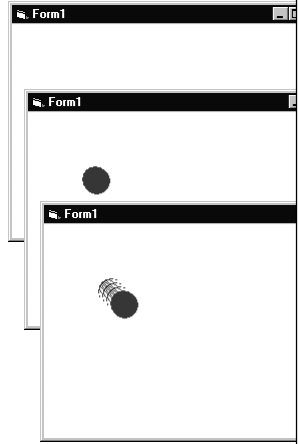
- ❖ Concept: Animation is simply the rapid display of still pictures
- ❖ The process --
 - ❑ Erase the figure
 - ❑ Reposition the figure by Δ distance
 - ❑ Redraw the figure
- ❖ Small Δ ==> slow, large Δ ==> fast
- ❖ Consider moving a circle across the form “with clicks”
 - ❑ Declare the position variables
 - ❑ Initialize them in Form_Load()
 - ❑ Apply the Process in Form_Click

© University of Washington, 2001

**FIT
100**

A Red Circle

❖ Code and Test the Process



```
Option Explicit
Dim xPos As Integer
Dim yPos As Integer
Private Sub Form_Click()
    Circle (xPos, yPos), 200, RGB(255, 255, 255)
    xPos = xPos + 20
    yPos = yPos + 20
    Circle (xPos, yPos), 200, RGB(255, 0, 0)
End Sub
Private Sub Form_Load()
    xPos = 1000
    yPos = 1000
    FillStyle = 0
    FillColor = RGB(255, 0, 0)
End Sub
```

© University of Washington, 2001

**FIT
100**

Indexing, A Basic Idea

- ❖ Motivation: When there is a large number of similar things that must be referenced and manipulated, it can be inconvenient to think up a unique name for each, and to refer to them by the name

+ For example: Each of the Seven Dwarfs has a name, but who can remember them? Also, it is difficult to refer to them in a loop since there is no way to enumerate them

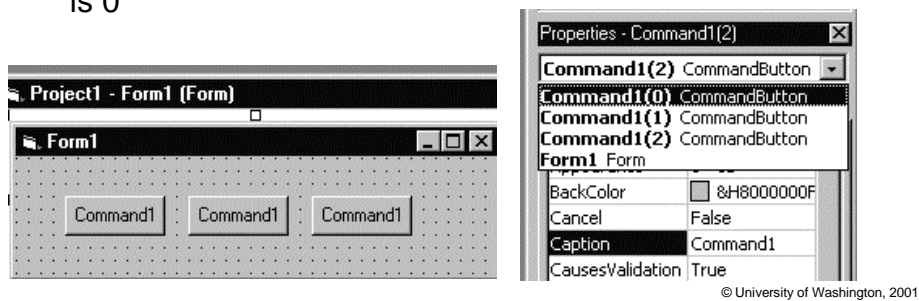
7 Dwarfs
Sneezy
Dopey
Sleepy
Grumpy
Happy
Doc
Bashful

- ❖ Concept: Indexing names items by associating a base name and a number -- the index -- with each
- ❖ Computer notation: Dwarf(5) ⇔ Happy

**FIT
100**

Indexing Particulars

- ❖ Everyday indexing commons begins with 1, e.g. May 1, SuperBowl I, Elizabeth I
- ❖ The number at which indexing begins is its *origin*
- ❖ Many computer languages use 1 as the origin, but for many others, including Visual Basic 6.0, index origin is 0

**FIT
100**

Arrays

- ❖ When a variable is indexed it is called an *array*
- ❖ Arrays represent collections of data values, e.g. integers, strings, etc.

For example: `dwarf(0) = "Sneezy"`

`dwarf(1) = "Dopey"`

`dwarf(2) = "Grumpy"`

...

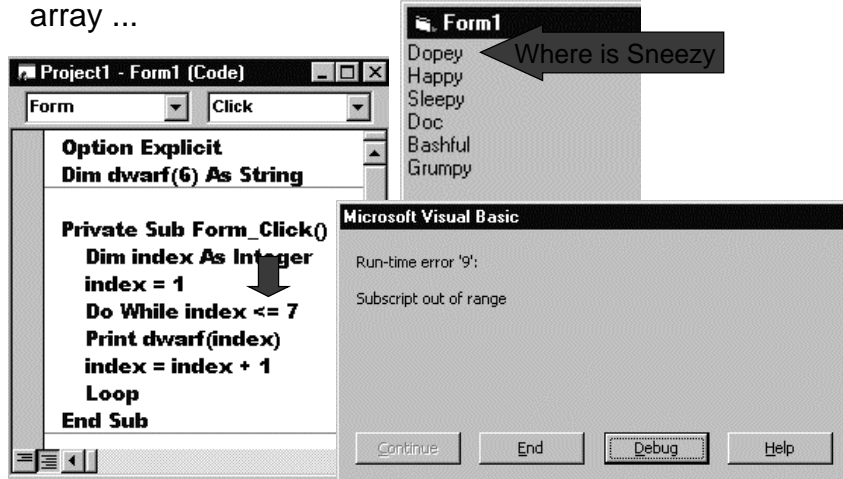
- ❖ Elements of an array must all be of the same type
- ❖ The index of an array element is also known as a *subscript*

Notice `x0` and `x1` are variable names, while `x(0)` and `x(1)` are different elements of array `x`

**FIT
100**

Combining Indexing, Arrays, Loops

- ❖ A common error is to index beyond the end of the array ...

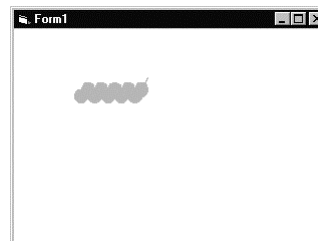


© University of Washington, 2001

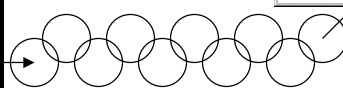
**FIT
100**

Practice Using Arrays

- ❖ Draw a 10-segment "inch worm" on the screen and move it forward
- ❖ Use arrays to keep the positions of the segments
- ❖ Write procedures to initialize worm and draw it
- ❖ Goals of exercise:
 - ❑ Practice with arrays
 - ❑ Practice with indexing
 - ❑ Practice writing procedures
 - ❑ Notice how arrays are passed as parameters



Center
inWx(0)
inWy(0)



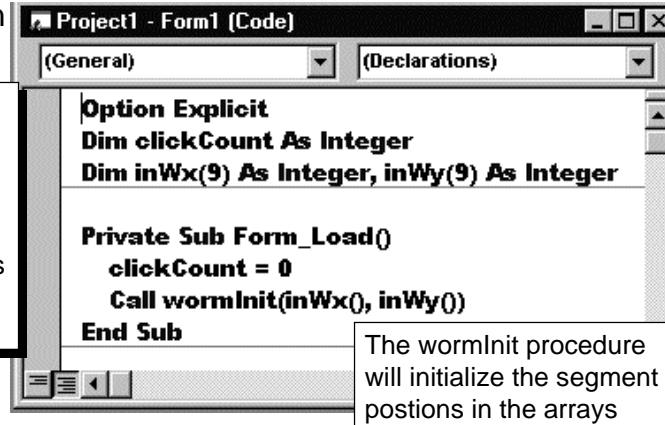
© University of Washington, 2001

**FIT
100**

Programming Drawing -- Step 1

- ❖ The first step is to declare the arrays and variables
- ❖ Will use the form click event handler to control the operation

The array inWx will store the x-coordinates for the segments, and inWy will store the y values. There are 10 segments



```
Project1 - Form1 (Code)
(General) (Declarations)

Option Explicit
Dim clickCount As Integer
Dim inWx(9) As Integer, inWy(9) As Integer

Private Sub Form_Load()
    clickCount = 0
    Call wormInit(inWx(), inWy())
End Sub
```

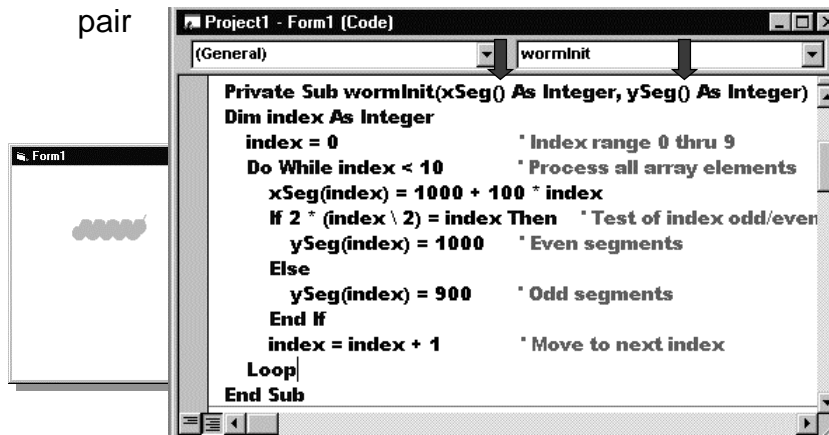
The wormInit procedure will initialize the segment positions in the arrays

© University of Washington, 2001

**FIT
100**

Step 2, Initialize The Coordinates

- ❖ When arrays are passed to procedures, the formal parameter must show this with an empty parenthesis pair



```
Project1 - Form1 (Code)
(General) wormInit

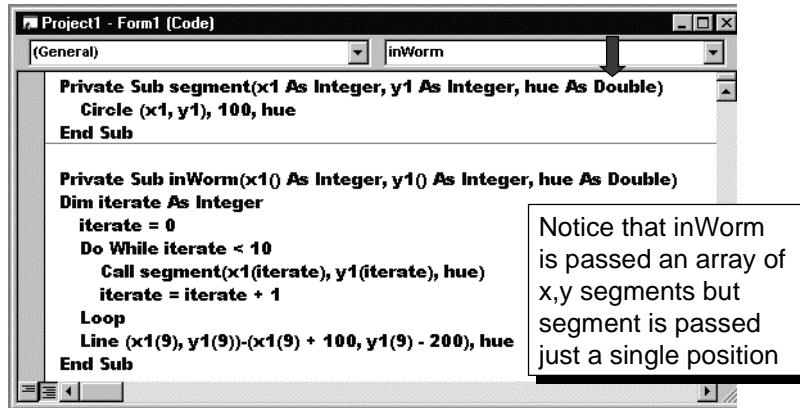
Private Sub wormInit(xSeg() As Integer, ySeg() As Integer)
    Dim index As Integer
    index = 0
    Do While index < 10
        xSeg(index) = 1000 + 100 * index
        If 2 * (index \ 2) = index Then
            ySeg(index) = 1000
        Else
            ySeg(index) = 900
        End If
        index = index + 1
    Loop
End Sub
```

© University of Washington, 2001

**FIT
100**

Step 3 -- Draw The Figure

- ❖ It will be necessary to draw the inch worm in different colors, so the color becomes a parameter -- Double



```
Project1 - Form1 (Code)
(General) inWorm

Private Sub segment(x1 As Integer, y1 As Integer, hue As Double)
    Circle (x1, y1), 100, hue
End Sub

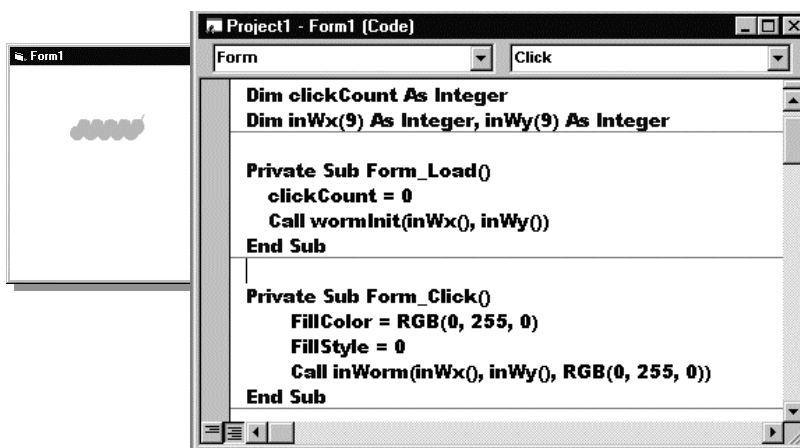
Private Sub inWorm(x1() As Integer, y1() As Integer, hue As Double)
    Dim iterate As Integer
    iterate = 0
    Do While iterate < 10
        Call segment(x1(iterate), y1(iterate), hue)
        iterate = iterate + 1
    Loop
    Line (x1(9), y1(9))-(x1(9) + 100, y1(9) - 200), hue
End Sub
```

Notice that inWorm is passed an array of x,y segments but segment is passed just a single position

© University of Washington, 2001

**FIT
100**

Draw On Click



```
Project1 - Form1 (Code)
Form Click

Dim clickCount As Integer
Dim inWx(9) As Integer, inWy(9) As Integer

Private Sub Form_Load()
    clickCount = 0
    Call wormInit(inWx(), inWy())
End Sub

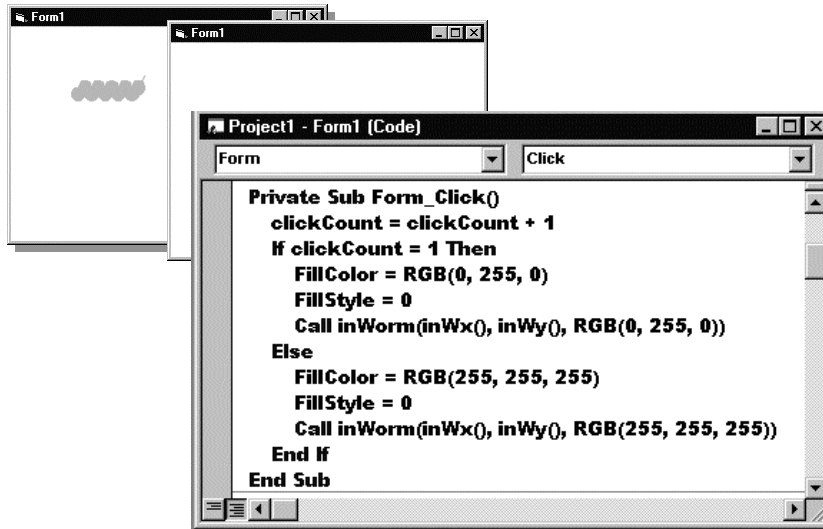
Private Sub Form_Click()
    FillColor = RGB(0, 255, 0)
    FillStyle = 0
    Call inWorm(inWx(), inWy(), RGB(0, 255, 0))
End Sub
```

To make the worm move, it must be erased/redrawn

© University of Washington, 2001

**FIT
100**

Click To Draw, Click To Erase

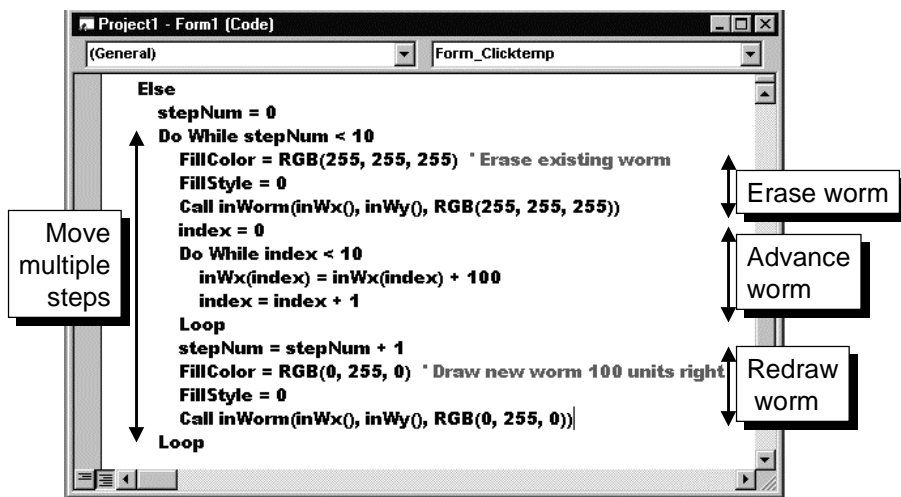


The screenshot shows a Visual Basic IDE with two windows. On the left, a window titled 'Form1' displays a simple drawing of a worm. On the right, a window titled 'Project1 - Form1 (Code)' shows the code for the 'Click' event of the 'Form' control. The code is as follows:

```
Private Sub Form_Click()  
    clickCount = clickCount + 1  
    If clickCount = 1 Then  
        FillColor = RGB(0, 255, 0)  
        FillStyle = 0  
        Call inWorm(inWx(), inWy(), RGB(0, 255, 0))  
    Else  
        FillColor = RGB(255, 255, 255)  
        FillStyle = 0  
        Call inWorm(inWx(), inWy(), RGB(255, 255, 255))  
    End If  
End Sub
```

**FIT
100**

Move Worm With Repeated Drawing



The screenshot shows a Visual Basic IDE with a code window titled 'Project1 - Form1 (Code)'. The code is for the 'Form_Clicktemp' event and includes a loop to move a worm. The code is as follows:

```
Else  
    stepNum = 0  
    Do While stepNum < 10  
        FillColor = RGB(255, 255, 255) ' Erase existing worm  
        FillStyle = 0  
        Call inWorm(inWx(), inWy(), RGB(255, 255, 255))  
        index = 0  
        Do While index < 10  
            inWx(index) = inWx(index) + 100  
            index = index + 1  
        Loop  
        stepNum = stepNum + 1  
        FillColor = RGB(0, 255, 0) ' Draw new worm 100 units right  
        FillStyle = 0  
        Call inWorm(inWx(), inWy(), RGB(0, 255, 0))  
    Loop
```

Annotations with arrows point to specific parts of the code:

- 'Move multiple steps' points to the 'Do While stepNum < 10' loop.
- 'Erase worm' points to the 'Call inWorm(inWx(), inWy(), RGB(255, 255, 255))' line.
- 'Advance worm' points to the 'inWx(index) = inWx(index) + 100' line.
- 'Redraw worm' points to the 'Call inWorm(inWx(), inWy(), RGB(0, 255, 0))' line.



Summary

- ❖ Animation is the repeated redisplay of still images
- ❖ Indexing is a general means of naming like things by a base name and a number
- ❖ Controls, variables and other objects can be indexed
- ❖ VB uses index origin 0