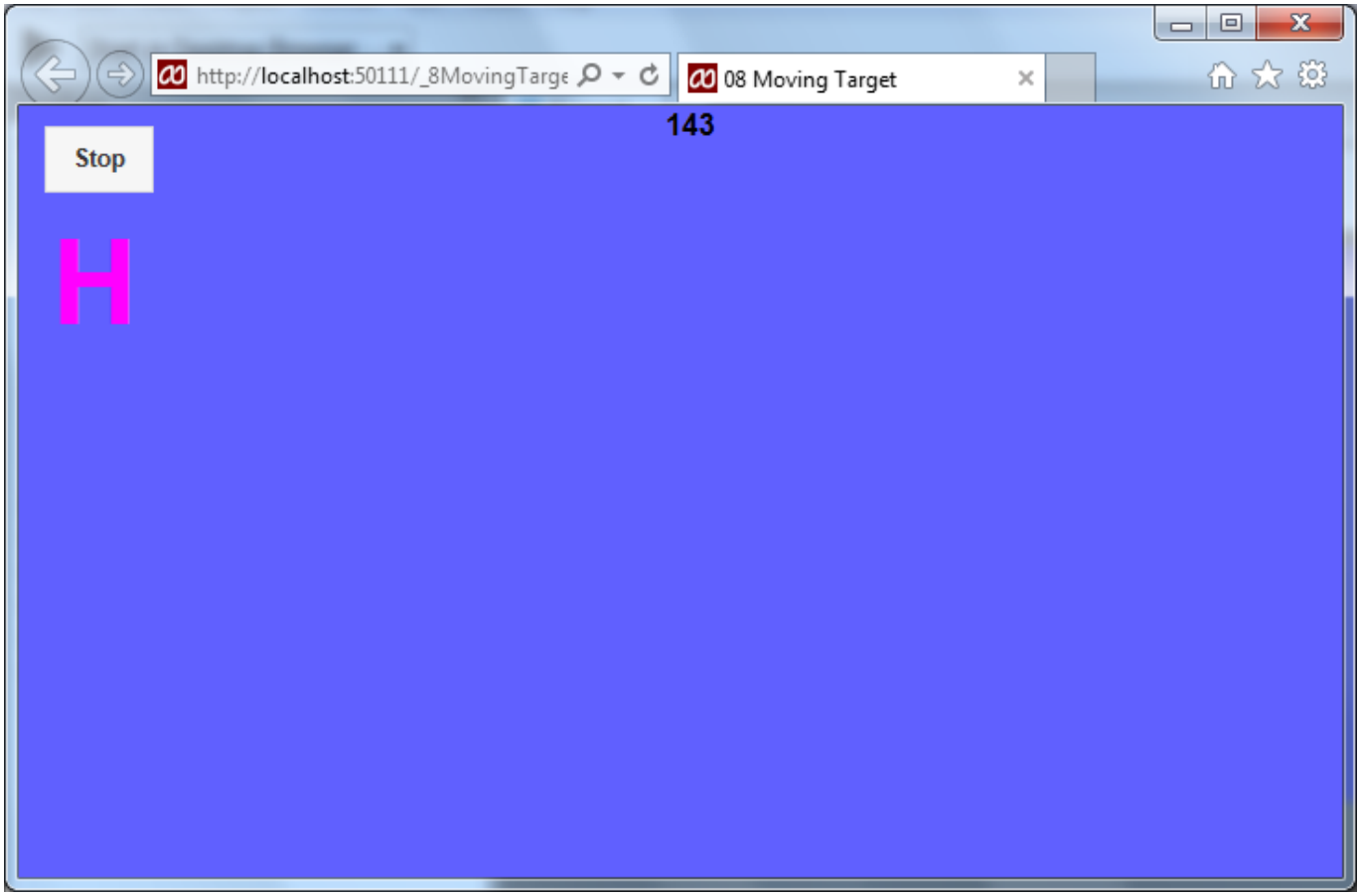
08 NSBasic – Moving Target

Download the Moving Target V3.00 NSBasic start file from your home folder. This is the file that you will start with in the end of year Exam, but the game in the Exam will develop in a different way, but some parts will be very similar!

If you run the file you will see a flashing pink "H" and a Timer that counts up to 300 in 30 seconds…



Try pressing Stop.

Version 3.01

The file has three image objects (Image1, Image2 & Image3). However they do not have any image. You need 3 Image files. Ideally png format, but they can be jpeg or gif. They need to be 44 x 44 pixels in size. The task of creating these will have been set as a homework. Below are ones that have been created. Do not use these, but ones that are appropriate for your website!

Add these to the folder where your NSBasic program is stored (Do remember to always use and check that you are using the P drive)! Then add these images to the three Image objects in Moving Target. Test the program fully to workout what it does!

Version 3.02

We will explore most of the code in the various developments in this workbook.

Label3 is the counter as to how much time has elapsed. The present game lasts 30 seconds. The counter counts up to 300. At the end of the Game it displays "Go over!" Find the code where "Go Over" is a change it to read "Game Over."

Find this code…

`i = i + 1`
`Label3.textContent = CStr(i)`
`If i >= 300 Then GameOver()`
	This updates the counter with i = i + 1,
	the next line displays the counter and the
	final line tests to see if the counter has reached the GameOver point.

What happens if we change the first line to i = i + 2 ?

Undo the change above, that is back to adding just 1 and change the If statement so that it ends after 10 seconds.

What happens if we change CStr(i) to Cstr(i / 10)? This is better as we now show the seconds more accurately.

An another alternative is `Label3.textContent = CStr(CInt(i / 10))` .

The CInt removes the decimal Part.

Now try… `Label3.text = "Seconds: " + CStr(CInt(i / 10))`

You might need to move the label to the left and make it a bit wider.

Version 3.03

We are now going to extend the gameplay by adding a “Power Play”. That is a short space of time when higher points can be awarded!

The Power play will be denoted by a P, rather than an H appearing.

First of all we need to make a “P” appear and test that we can score double points.

Look in **`Function`** `HitNow()` where you will see 2 H’s change each H to a P.

If you run the program now you will not score any points, but a P will appear.

Now we need to score double points for hitting Image1 when in P mode!

Look in `Function Image1_onclick()` This is the code that is carried out or executed when you click on Image1. The code looks to see if an H is being displayed and then adding a point on. We could change the code to…

```
Function Image1_onclick()
  If Label1.textContent = "P" Then
    Score = Score + 2
  Else
    Score = Score - 2
  End If
  Label2.textContent = Score
  Image1.Left = Border + Int(Rnd()*
```

Yet this removes any points for an hitting when an H is displayed.

A new logic is needed… Let’s assume that the player hits when they should not hit the target and so 2 points need to be deducted. That is

```
Dim HitPoints
HitPoints = -2
```

Now we can add 2 lines to test for the P or H….. at which point Score becomes 2 or 1 respectively..

```
If Label1.text = "P" Then HitPoints = 2
If Label1.text = "H" Then HitPoints = 1
```

Now all we need to do is to update the score (add HitPoints) and display it in Label2…

```
Score = Score + HitPoints
Label2.text = Score
```

Now the fully revised Function…

```
Function Image1_onclick()
  Dim HitPoints
  HitPoints = -2
  If Label1.textContent = "P" Then HitPoints = 2
  If Label1.textContent = "H" Then HitPoints = 1
  Score = Score + HitPoints
  Label2.textContent = Score
  Image1.Left = Border + Int(Rnd()* (Form1.Width - (Label1.Width + Border)))
  Image1.Top = Border + Int(Rnd()* (Form1.height - (Label1.height + Border)))
End Function
```

The last 2 lines are unchanged – What do they do?

Answer: (Reveal by blocking your home folder version of this file)

[redacted]

Label2 could be made to be a little more helpful by adding the word “Score:”, before displaying it…

```
Label2.text = "Score: " + CStr(Score)
```

CStr is added as we need to convert the number in Score to be placed after the text inside the double quotes!

Now we need to make both P & H appear at random.

Move back to **Function** HitNow() where we will need to develop this code….

```
Function HitNow()
  HitNow = Label1.textContent
  Select Case HitNow
    Case ""
      If Rnd() < 0.2 Then HitNow = "H"
      If Rnd() < 0.1 Then HitNow = "P"
    Case "H", "P"
      If Rnd() < 0.1 Then HitNow = ""
    Case Else
      HitNow = ""
  End Select
```

This will now change to a blank if H or P are displayed by the odds of 0.1 = 1 / 10.

It will Change from a blank to an H 1 time in 5 (0.2 = 1/5), However there is a 1 in 10 chance that it will change to a “P”.

So “P” is less likely to be displayed, yet can remain as long as an H! We need to shorten this time…

So a further enhancement is…

```
Function HitNow()
  HitNow = Label1.textContent
  Select Case HitNow
    Case ""
      If Rnd() < 0.2 Then HitNow = "H"
      If Rnd() < 0.1 Then HitNow = "P"
    Case "H"
      If Rnd() < 0.1 Then HitNow = ""
    Case "P"
      If Rnd() < 0.25 Then HitNow = ""
    Case Else
      HitNow = ""
  End Select

End Function
```

As now P will turn back to a blank sooner!

Version 3.04

In the Powerplay you can score 1 point for hitting Image2. You do not loose points for hitting this at the wrong time… Much of the code exists in `Function Image1_onclick()` …
This time you need a `Function Image2_onclick()` This can be copied, pasted and changed…

```
Function Image2_onclick()
  Dim HitPoints
  HitPoints = 0
  If Label1.textContent = "P" Then HitPoints = 1
  Score = Score + HitPoints
  Label2.textContent = "Score: " + CStr(Score)
  Image2.Left = Border + Int(Rnd()* (Form1.Width - (Label1.Width + Border)))
  Image2.Top = Border + Int(Rnd()* (Form1.height - (Label1.height + Border)))
End Function
```

Do make sure you change the last 2 lines from Image1 to Image2… Otherwise what might happen?? Answer: (Reveal by blocking your home folder version of this file)

[redacted]

You need to check that the Image2 property Onclick is set to Image2_onclick()

Version 3.05

If you hit Image3 at any stage you should lose 5 points! Workout the code for yourself that will do this.

Answer: (Reveal by blocking your home folder version of this file)

[redacted]

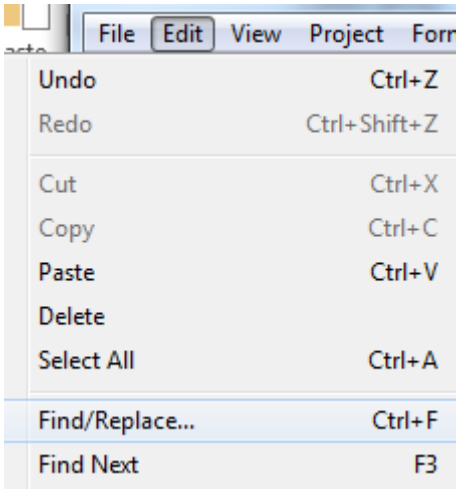
Version 3.06

Image2 should only be displayed when the Score is greater than 1. This is harder to achieve as the Score can change in various places. To solve this we must first move the Score updating into a common place. That is we should give it its own Subroutine …

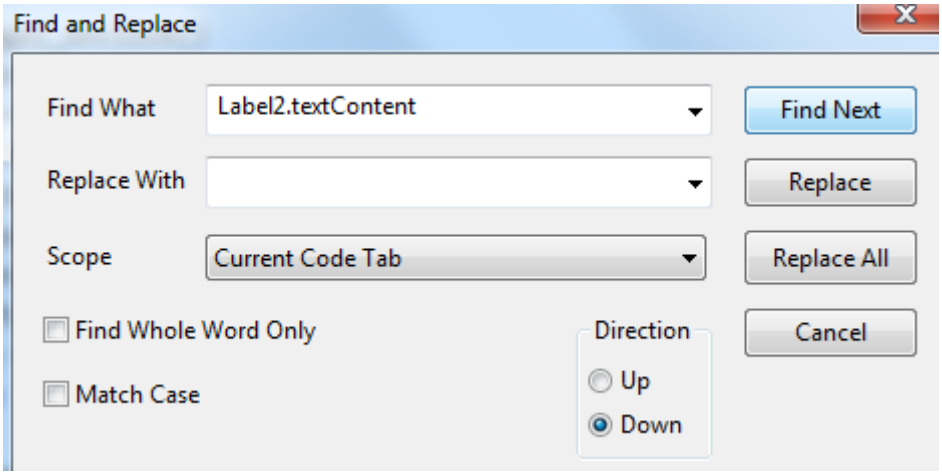
```
Sub DisplayScore()
  Label2.text = "Score: " + CStr(Score)
End Sub
```

Now we need to find all the lines where the Display of Label2 is changed…

In NSBasic move to the top (line 1) of your code, then go to the Edit Menu and scroll down to Find/Replace



This window appears..



Enter the Find What text as shown above and click “Find Next” (do not use Replace or Relace All!)

Manually change the whole line in the program window from….

```
Label2.text = "Score: " + CStr(Score)
```

to

```
DisplayScore()
```

But do not change it inside the Subroutine DisplayScore!!

Repeat the Find option until all entries have been changed.

Test that the program still works.

As we now have all the code in one place we can enhance the code in many ways.

The aim of this version was to make Image2 appear / disappear based on the Score.

The Code for DisplayScore() can be developed further…

The code below is wrong! It has all of the correct statements, but they are in the wrong location…

```
Sub DisplayScore()
  If Score > 1 Then
    Image2.hidden = True
    Image2.hidden = False
  Else
    Label2.textContent = "Score: " + CStr(Score)
    End If
End Sub
```

Workout the correct order and Image2 will appear and disappear according to the Score!

Version 3.07

There is at least one bug! This has been there since version 3.00. We now need to fix some of the bugs. Yet first you need to understand the problem(s). Play the game in version 3.06 4 or 5 times. Try to gain 2 or more top scores. That is the first game aim for a TopScore of 1 or 2. Then it a later game aim for 3 or higher. You can "Hold" a top score by pressing stop once it has been achieved. If you have an older version, try the same test. Older versions may have the same problem or that the code works or a problem in-between! You will notice a number of problems, but there are 2 we will focus on at this stage…

Hopefully you will have seen that the Score does not reset when a new game starts. Often problems are one or two lines of faulty code near one another. Other times the problem is more widespread.

This problem needs to be fixed in two places. The code is not used consistently. The start of the game and at the re-start there are lots of similar tasks that need to be reset. Then there are parts which should not be reset (like the Top Score). The present code uses almost identical code in 2 locations. This is a poor practise and we will improve it… Look at the code below…

Version 3.06 code

```
Function Button1_onclick()
  If Button1.value = "Start" Then
    i=0
    Button1.value = "Stop"
    timerRef = SetInterval(nextAction, 100)
  Else
    Stop()
  End If
End Function

Sub StartUp()
  i = 0
  Button1.value = "Stop"
  timerRef = SetInterval(nextAction, 100)
  DisplayScore()

End Sub
```

New Code (v3.07)

```
Function Button1_onclick()
  If Button1.value = "Start" Then
    StartUp()
  Else
    Stop()
  End If
End Function

Sub StartUp()
  Score = 0
  i = 0
  Button1.value = "Stop"
  timerRef = SetInterval(nextAction, 100)
  DisplayScore()
End Sub
```

In the new code Button1 simply calls StartUp or Stop. StartUp is extended to reset Score with a new first and last line! StartUp does not change the TopScore which Label4 is used for.

Try the earlier test again. Does the score now reset to zero at the start of everygame? It should do!

Now for the second problem… TopScore … If you had saved older versions then you may have noticed that this partially worked. However version 3.06 made the problem worse. **It is important to remember that changing one piece of code may have an adverse effect on another part of the program.** This is what has happened here and it represent a good example of it!

Now to fix this "glitch"…

Version 3.06 code

```
Sub GameOver()
  ClearInterval(timerRef)
  Label3.textContent = "Game over."

  If CInt(Label2.textContent) > TopScore Then
    TopScore = CInt(Label2.textContent)
    Label4.textContent = "Top Score is " + TopScore
    Label4.hidden = False
  End If
  Button1.value = "Start"
End Sub
```

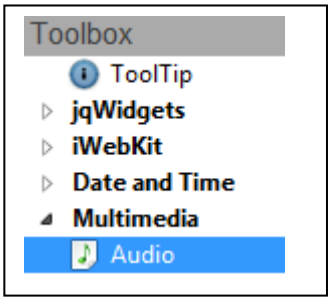
New Code (v3.07)

```
Sub GameOver()
  ClearInterval(timerRef)
  Label3.textContent = "Game over."

  If Score > TopScore Then
    TopScore = Score
    Label4.textContent = "Top Score is " + TopScore
    Label4.hidden = False
  End If
  Button1.value = "Start"
End Sub
```

Version 3.06 used the displayed information, rather than the "global variable" TopScore. Hence when we changed the display we stopped the code from working. It is always better to use the variable and then output that with any formatting, in this case the words "Top Score is". It is worth noting that TopScore will not appear until a TopScore is set which is greater than 0. Why, well TopScore is set to 0 and hence the Label4.hidden = False will only happen when a new TopScore is found. TopScore is initially 0, so no negative TopScores will ever be displayed!

Version 3.08



Sounds could be added at various points. Though it is always helpful to have a way to turn these off or down to a minimal amount! Add Three Audio objects from the Toolbox each with a different and quite short sound (1 second or less). The mp3 file needs to be copied into the folder that has the program and images. Then you set the src property to the file (just like with images). Now you can choose when to play them with the objectname and .play() ie Audio1.Play()
You should experiment with the sounds.

Add a fourth and slightly longer, cheerful / successful sound. This should play when a new TopScore is created. In this Workbook only Audio4.play() code has been added from this point on…

Version 3.09

To make the game a little easier at the start… The main Target object (Image1) will only move up and down across the screen. To do this look at Sub Setup()

Version 3.08 code

```
Sub SetUp()
  AImage(1)=Image1
  Movehorizontal(1) = 2
  Movevertical(1) = 6

  AImage(2)=Image2
  Movehorizontal(2) = 5
  Movevertical(2) = 3

  AImage(3)=Image3
  Movehorizontal(3) = -1
  Movevertical(3) = 6
End Sub
```

New Code (v3.09)

```
Sub SetUp()
  AImage(1)=Image1
  Movehorizontal(1) = 0
  Movevertical(1) = 6

  AImage(2)=Image2
  Movehorizontal(2) = 5
  Movevertical(2) = 3

  AImage(3)=Image3
  Movehorizontal(3) = -1
  Movevertical(3) = 6
End Sub
```

There is only one number that has changed. Try it and see.
What happens if you change the line Movevertical(1) = 6 to Movevertical(1) = 0 ?

Instead of the 2 zeros change one of the zeros to 1, then to 2, then to 5 then to 10 then to 20.

Reset it to zero and try the same test with the other one. Try negative numbers too!

Carryout these tests on… Movehorizontal(2) / Movevertical(2) then Movehorizontal(3) / Movevertical(3).

You should now know what they control. Check the Answer below….

Start the game with the main target Image1 not moving.

Version 3.10

Once the score reaches 5, we can now start the movement of Image1.

Sub DisplayScore was set up in version 3.06, now we can use it to change the speed of Image1…

Add this code before the End Sub

```
If Score > 4 Then
  Movehorizontal(1) = 3 : Movevertical(1) = -2
Else
  Movehorizontal(1) = 0 : Movevertical(1) = 0
End If
```

Now test to see what happens…

Version 3.11

It would be good if the speed of Image1 moved quicker as the score progressed. This change helps with this… The code added in 3.10 is changed to….

```
If Score > 4 Then
  Movehorizontal(1) = 3 + Score : Movevertical(1) = Score - 2
Else
  Movehorizontal(1) = 0 : Movevertical(1) = 0
End If
```

As this code is carried out everytime the score changes, then it will slow down as and when points are deducted. This code is not ideal as the start direction is quite fixed for each score. It can be improved, but that involves some complex mathematical formula.

Version 3.12

The skill of playing this game is about watching two different parts of the screen. The target and the "Hit" status. Presently only the Target moves. If the Score goes above 10 a new level could be reached, where the Hit Status moves to another fixed place on the screen.

Again above the End Sub of Sub DisplayScore() We can add this code…

```
  If Score > 10 Then
  Label1.Left = 200
Else
  Label1.Left = 16
End If
```

However to test this we need to wait until a score of 10 is reached. This may be hard to achieve. So instead of typing in 10, type in 1. That is it will work once the score reaches 2 or more. It also moves to a fixed left to right position.

Test it to see if it works.

Now we can develop it further.

See if you can move it Down to a lower position as well as moving right….

Hint: [redacted]

Now can we move it to a random position…

Part of the code for this already exists… within Image1_OnClick()

```
.Left = Border + Int(Rnd()* (Form1.Width - (Label1.Width + Border)))
```

Use this instead of 200.

Find the equivalent line to move it to a random position Up and down.

After testing reset the IF statement back to 10. Test once again.

Version 3.13

We are now going to add 2 additional moving Images, though one at a time!

Add Image4 by dragging the item onto the form. Make it the same size as Images1 – 3 (44 x 44). Add a new graphic for image4.

Change the Dim Statement at the very top of the code to read…

```
Dim NumberOfImages = 4
```

Then at the bottom of the code in Sub SetUp() add just before the end Sub…

```
AImage(4)=Image4
Movehorizontal(4) = 3
Movevertical(4) = -2
```

Now test. The fourth Image should be present and moving.

Repeat the above but for Image5 (in theory this could be done for Image6 / Image7 etc.!

However where there is a 4 above change it to a 5. The values 3 and -2 which from earlier work you should recall are the speed and direction of movement can be changed as well (set both to 0 if no movement is required initially).

Version 3.14

Image4 and Image5 are to Score 3 points when hit initially. At that point Image4 is to start moving, whilst Image5 is to freeze. They will then not be worth any more points!

Adjust Image4 initial speed in SetUp to make it stationary.

Now we need to add code for looking to see if it is hit by the game player…!

Copy and paste Function Image3_onclick() directly below itself…

```
Function Image3_onclick()
  Dim HitPoints
  HitPoints = -5
  Score = Score + HitPoints
  DisplayScore()
  Image3.Left = Border + Int(Rnd()* (Form1.Width - (Label1.Width + Border)))
  Image3.Top = Border + Int(Rnd()* (Form1.height - (Label1.height + Border)))
End Function
```

Change “Image3” to “Image4” in the three locations Image3 appears. Change HitPoints from -5 to 3.

Test to see if it works. That is that it does not move and if you hit it you score 3 points.

As it is based on Image3 after being hit it relocates itself. Three points can also be scored over and over again.

The idea for Image4 is that it starts moving after it has been hit the first time then at that point no more points can be scored. It has two states. We can test for which state with an IF statement and then write our code accordingly. The “test” we need to ask is “Is it moving?” This can be carried out by looking at the values of Movehorizontal(4) and Movevertical(4) which both need to be 0 if it is not moving. Hence…

```
Function Image4_onclick()
  If Movehorizontal(4)=0 And  Movevertical(4)=0 Then
    Dim HitPoints
    HitPoints = 3
    Score = Score + HitPoints
    DisplayScore()
    Movehorizontal(4) = 2 - Int(Rnd()*5)
    Movevertical(4) = 2 - Int(Rnd()*5)
  Else
  End If
  Image4.Left = Border + Int(Rnd()* (Form1.Width - (Label1.Width + Border)))
  Image4.Top = Border + Int(Rnd()* (Form1.height - (Label1.height + Border)))
End Function
```

Change your Function to the above, but think about what it will do.

There is a “bug” of sorts in this code. By bug, it will work but there is a chance that it will not quite work as planned. The bug is not the present pointless Else statement (That is to help with Image5 in the version below). It is the way the speed and direction are calculated through 2 – Int(Rnd()*5)… What does this give…

- Rnd() is a number between 0 & 0.9999.
- Multiplying by 5 gives numbers from 0 to 4.9997.
- The Int (for Integer = whole number part) changes the numbers to 0 / 1 / 2 / 3 / 4.
- 2 minus these numbers gives 2 – 0 = 2 to 2 – 4 = -2.

This includes 0! Both options have the potential to be 0 – which the IF statement tests for and will allow you to score 3 points again! The chance of each option being 0 is 1 in 5. Combined this is 1/5 x 1/5 = 1/25 or 1 in 25 chance. For the time being we will leave this as it can be a feature of the game!

Version 3.15

You can use the same ideas as in Version 3.14 for Image5. You are advised to leave the first line of IF statement as it is but move code between the Then / Else / End IF lines depending on its state and what Image5 is doing. To recap Image5 must initially move, but when hit the first time it stops moving. The first hit only scores 3 points.

Version 3.16

If Image5 is hit a 2nd time, then the Score is reset to zero and Image5 disappears!

You will need three lines of code to follow the Then part of the IF statement

The first hides Image5

The next two set the main Score to 0 and Display the Score.

Add these lines. If struggling after 5 minutes reveal the Hint below…

```
If Movehorizontal(5)=0 And  Movevertical(5)=0 Then
[redacted]
Else
```

Version 3.17

We can use variations of 2 – Int(Rnd()*5) in the SetUp() or StartUp() routines to set different speeds and directions for the Images.

Presently the initial values are set in SetUp. Yet they need to be reset whenever a new game commences and they presently keep there old values. This is now a bigger problem and some images (4 & 5) can change their status in the course of a game.

We need to move some of the code from SetUp to StartUp.

So Setup becomes just…

```
Sub SetUp()
  AImage(1)=Image1
  AImage(2)=Image2
  AImage(3)=Image3
  AImage(4)=Image4
  AImage(5)=Image5
End Sub
```

Whilst the code now appears at the top of StartUp..

```
Sub StartUp()
  Movehorizontal(1) = 0
  Movevertical(1) = 0

  Movehorizontal(2) = 15
  Movevertical(2) = -30

  Movehorizontal(3) = -10
  Movevertical(3) = 10

  Movehorizontal(4) = 0
  Movevertical(4) = 0

  Movehorizontal(5) = 2
  Movevertical(5) = 1

  Score = 0
  i = 0
```

Some Images are supposed to be hidden at the start of the game, whilst others can be hidden during the game itself. These too need to be reset. Add the appropriate code like ImageX.hidden = True / False above the Score = 0 line.

Test that it all works for Game1 and Game 2 onwards…!

Version 3.18

We can now use variations of 2 – Int(Rnd()*5) in StartUp. Before we use this we can introduce a loop to help us…

Modify the start of StartUp to that shown.

The Loop counter L starts at 1 and moves through all of the Images to the NumberOfImages in our program.
The speed and direction of each image is fixed to 2 / -3
The Next statement concludes the For Loop

Images 1 & 4 have their movement removed. The Colon allows 2 lines to be joined.

The hidden status of two images is also reset for the beginning of each game.

The rest of the code runs as before with Score being reset to zero.

```
Sub StartUp()
  For L = 1 To NumberOfImages
    Movehorizontal(L) = 2
    Movevertical(L) = -3
  Next

  Movehorizontal(1) = 0 : Movevertical(1) = 0
  Movehorizontal(4) = 0 : Movevertical(4) = 0

  Image2.hidden = True
  Image5.hidden = False

  Score = 0
  i = 0
```

Now we can substitute in the code 2 - Int(Rnd()*5) instead of the 2 and -3 within the For loop. This randomly gives each Image a different speed and direction. Yet it has the problem that was mentioned earlier that it is possible for both to be zero and the image will not move.

Test that it works.

We could change the 2 to a 3 and the 5 to a 7. This will cause a greater variety of speeds and reduce the chance of an additional static object. You can experiment with the first number going up in ones and the second by 2's ie the following pairs… (2,5) (3,7) (4,9) (5/11) (6,13) So the last could be… code 6 - Int(Rnd()*13)

Version 3.19

Now we can randomly set the start position as well as the speed of the objects.

To do this we extend the loop from the last version…

```
Sub StartUp()
  For L = 1 To NumberOfImages
    Movehorizontal(L) = 4 - Int(Rnd()*9)
    Movevertical(L) = 4 - Int(Rnd()*9)
    AImage(L).Left = Border + Int(Rnd()* (Form1.Width - (Label1.Width + Border)))
    AImage**************************************************************************
  Next

  Movehorizontal(1) = 0 : Movevertical(1) = 0
```

Now the correct line before the Next will need some thinking. See if you can work it out!

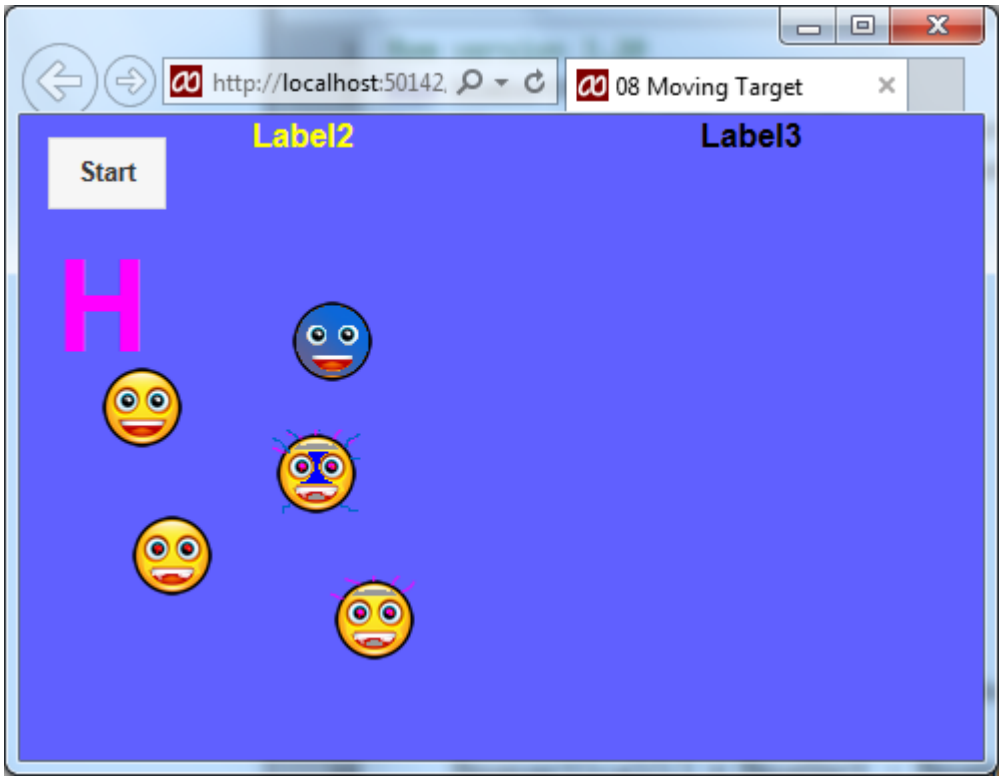
If after 10 minutes you are still struggling use the hint below…

```
AImage
```

Version 3.20

The game presently starts as soon as it loads. It might be better for the first action from the Game player to be pressing on Start. To make this change the text in the Button to Start and rem out the StartUp() line at the top of the program code. Directly after SetUp().

The problem now is…



Label2 and Label3 appear and although the images do not move they can score points!

This can be fixed by pasting in the following code…

```
Sub SetUp()
  AImage(1)=Image1
  AImage(2)=Image2
  AImage(3)=Image3
  AImage(4)=Image4
  AImage(5)=Image5
  For L = 1 To NumberOfImages
    AImage(L).hidden = True
Next
Label1.hidden = True
Label2.text = "Press Start for the First game to commence.."
  Label3.hidden = True
  Label4.hidden = True
End Sub
```

As a replacement to the present SetUp

And by changing the first few lines to…

```
1  Rem version 3.20
2  Dim NumberOfImages = 5
3  Dim AImage(NumberOfImages),Movehorizontal(NumberOfImages), Movevertical(NumberOfImages)
4  Dim timerRef, i, TopScore, Score, MoveHoriz, MoveVert, Border
5
6  TopScore = 0
7  Score = 0
8  Border = 10
9  SetUp()
10
11 Function nextAction()
12   Label1.textContent = HitNow()
```

Test and you will see that we need to extend StartUp to redisplay the parts we have hidden!

The line numbers below do not matter, but make sure you have the code in sequence as shown below… The key line numbers with the code that has been changed are 65, 66 & 69.

```
64 Sub StartUp()
65 Label1.hidden = False
66 Label3.hidden = False
67
68   For L = 1 To NumberOfImages
69     AImage(L).hidden = False
70     Movehorizontal(L) = 4 - Int(Rnd()*9)
71     Movevertical(L) = 4 - Int(Rnd()*9)
72     AImage(L).Left = Border + Int(Rnd()* (Form1.Width - (Label1.Width + Border)))
73     AImage(L).Top = Border + Int(Rnd()* (Form1.height - (Label1.height + Border)))
74   Next
75
76   Movehorizontal(1) = 0 : Movevertical(1) = 0
77   Movehorizontal(4) = 0 : Movevertical(4) = 0
```

Version 3.21

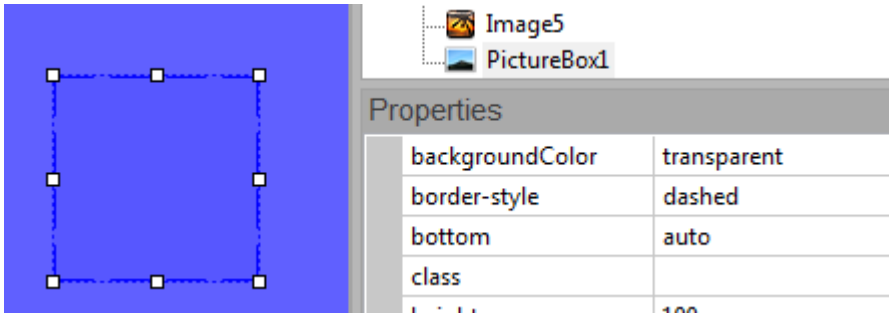
The code you worked out in version 3.19 for the Left position is quite complex. What it does is to workout a position within the present window and inside a Border setting. The Border setting can be adjusted as it is set to 10 in the first few lines of the code. Try different values (between 0 & 50) to see what happens…

[redacted]

Check the line above to see if you were correct.

Now it would be good to see the “bounds” of the border. To do this we will introduce a PictureBox.

Set the Properties as shown…



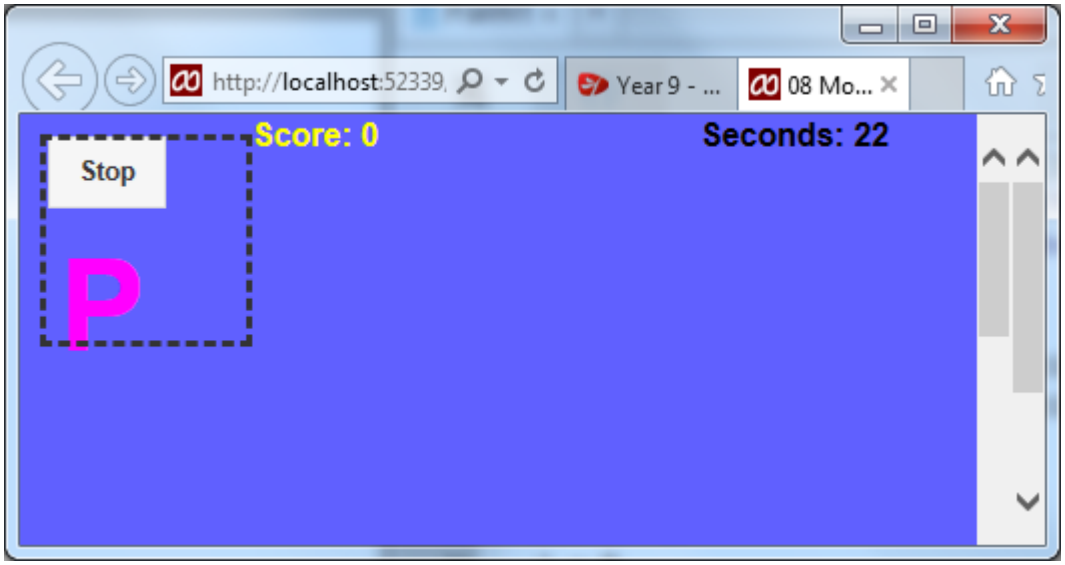
We will now use code to determine where the box should be placed. Inside SetUp() we must make it hidden (add this before the end Sub)…

```
        PictureBox1.hidden = True
```

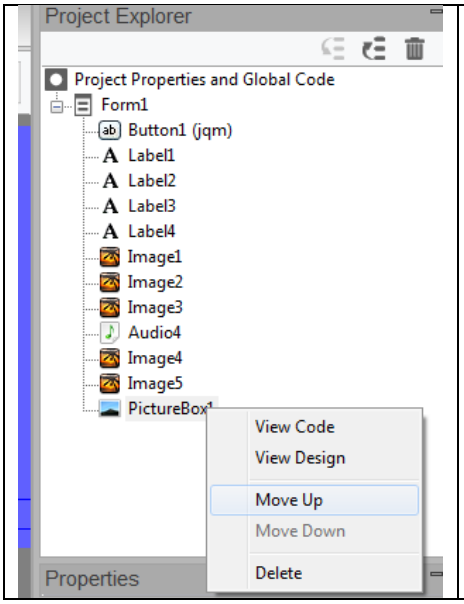
Then inside StartUp(), again before End Sub…

```
        PictureBox1.Left = Border
        PictureBox1.Top = Border
        PictureBox1.hidden = False
```

After pressing Start it should look like this…

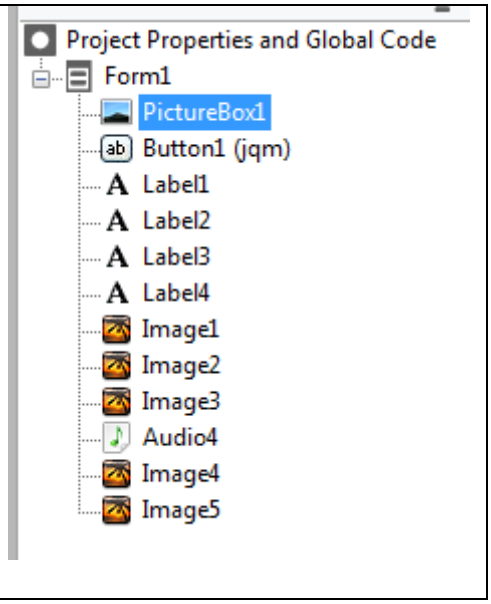


There is a problem as you cannot click on Stop! Try it and see! This is to do with the order the objects appear in.



We can fix this (wrong on the left) by…

Right Mouse and using Move Up Until it reaches the top as shown on the right…



Test and see that it works! Start / Stop should work again. Now we can workout where the bottom and right edge should be placed…

The code shown to the right should be at the bottom of StartUp()…

The Border is x 2 as this is doubled due to space lost at the top and bottom.

```
    DisplayScore()

    PictureBox1.Left = Border
    PictureBox1.Top = Border
    PictureBox1.Height = Form1.Height - (Border * 2)
    PictureBox1.Width = Form1.Width - (Border * 2)
    PictureBox1.hidden = False
End Sub
```

The size is calculated when each game starts.

Version 3.22

We now have a way to record a miss… If you click on PictureBox1, then all other objects have been missed. Hence we can deduct points for a miss very simply…

```
Function PictureBox1_onclick()
  Score = Score - 10
  DisplayScore()
End Function
```

The PictureBox outline also cuts through Labels 2, 3 & 4. This can be fixed by Changing Border to 20 in the first few lines of the code rather than the present value of 10.

The Start / Stop Button is also in the game area. This can be move by typing in new Property values for Top of 0, Height as 20 (from Auto) and a FontSize of 10.

Version 3.23

A PictureBox is different to an Image in the way we can display information inside it. We are able to write both text and images from the code we write…

A new global variable which will represent the paper we can draw on… At the top of the code ass pictbox1 to the Dim statement. Then below that (line 5 in example below) add the whole line…

```
pictbox1 = PictureBox1.getContext("2d")
```

```
4 Dim timerRef, i, TopScore, Score, MoveHoriz, MoveVert, Border, pictbox1
5 pictbox1 = PictureBox1.getContext("2d")
6 TopScore = 0
```

These 2 lines link a “piece of paper” that we can write or draw on to the PictureBox1 object.

Add these lines to the end of Function PictureBox1_onclick()
 	
`pictbox1.textBaseline = "top"`
`pictbox1.font="160px sans-serif"`
`pictbox1.fillText("Miss",100,100)`
 	
as shown on the right…
	`Function PictureBox1_onclick()`
	`  Score = Score - 10`
	`  DisplayScore()`
	`  pictbox1.textBaseline = "top"`
	`  pictbox1.font="160px sans-serif"`
	`  pictbox1.fillText("Miss",100,100)`
	`End Function`

Test this – If you Miss an object a large black Miss appears.

To change the Colour you can use the line

```
pictbox1.fillStyle = RGB(255,0,255)
```

Add this after pictbox1.textBaseline = "top"

You can change each part of the R (ed) G (reen) B (lue) component as long as it is in the range 0 to 255.

We now need a way to remove the “Miss”. We also need to consider when to remove it. If we remove the Miss when the Score is next updated.

Then we just place the code at the end of Sub DisplayScore(). That is on a new line directly before the End Sub..
 	
`pictbox1.clearRect(100, 100, 180, 200)`
	`    Label1.Top = 55`
	`End If`
	`pictbox1.clearRect(100, 100, 180, 200)`
	`End Sub`

Test to see if this works.

You will see that it partially works! The Mi are deleted but most of the ‘ss’ remain. That is because the size is not big enough. The 180 represents the width, whilst 200 is the height of the area being cleared. The 180 is not large enough whilst the 200 is too much. Try 380 & 120 instead. This time part of the bottom of the writing is left! Now workout the correct values to clear it all.

Version 3.24

It would be possible to display an Image behind the Word “Miss” within PictureBox1.

To do this create an image and store the image in your NSBasic folder that has your code and other images in.

If the file is KES001.jpg then you need this code…

```
pictbox1.addImage("KES001.jpg",100,0,380,150)
```

The numbers represent the Left and top corner followed by the image width and image height. Add this code to Function PictureBox1_onclick() directly after the DisplayScore() line.

You will need to change pictbox1.clearRect(100, 100, 180, 200) so that the image is also deleted.

Version 3.25

The Score should not be allowed to go below zero.

This can be achieved by adding an IF statement as a new first line to `Sub DisplayScore()` See if you can work it out – If not reveal the following code (try for 5 mins)…[redacted]

You may wish to resize the various labels so that the text does not move down to a 2nd line as well.

Version 3.26

When the game ends it is still possible to partially play on, even if the Top Score is not updated. So when the Start button displays ‘Start’ then all other clicks should be disabled. That means no code should be excuted.

To do this we can test the state of the Start button to see if it is set to Start or not.

For example… The first line tests to see If Button1 does not equal / is not displaying the value “Start”

```
Function PictureBox1_onclick()
  If Button1.value <> "Start" Then
  Score = Score - 10
  DisplayScore()
  pictbox1.addImage("KES001.jpg",100,0,380,150)
  pictbox1.textBaseline = "top"
  pictbox1.fillStyle = RGB(255,0,255)
  pictbox1.font="160px sans-serif"
  pictbox1.fillText("Miss",100,100)
  End If
End Function
```

< Note the addition of the End IF!

This should now be added to all Image (Image1 to Image5) onclick() routines. Do not forget the “End IF” as well!

Version 3.27

The Start Button could become larger when Start is required. We will need to make it smaller once pressed too!

Button1 is set to Start in two locations…. Now we need to use more than 1 line of code to manage all the changes. It is never a good idea to copy and paste code, it is much better to create a new (Sub) routine to do all the work for us and call that from the various locations.

Firstly we will create the subroutine then we will add the lines to call the subroutine

```
Sub Button1Display()
  $(Button1).css("font-size", "50px");
  Button1.resize(20,20,400,400)
  Button1.value = "Start"
End Sub
```

```
Sub Button1Display()
 $(Button1).css("font-size", "50px");
 Button1.resize(20,20,400,400)
 Button1.value = "Start"
End Sub
```

This code changes the font size to 50, then resizes the box, before changing the display to read Start.

Now replace the two lines of `Button1.value = "Start"` with `Button1Display()`

Test to see what happens!

The problem now is that it does not reset when it changes to “Stop”!

Find the line… `Button1.value = "Stop"` it only appears once!

	Add this code on the lines directly before it.

`$(Button1).css("font-size", "10px");`
`Button1.resize(14,0,59,20)`
`Button1.value = "Stop"`
`timerRef = SetInterval(nextAction, 100)`
	$(Button1).css("font-size", "10px");
	Button1.resize(14,0,59,20)

Now we just need to set the correct size when the program starts. So relocate the Form on the design page of NSBasic.

Version 3.28

As the game player progresses the game needs to become harder! Many of the changes have made the game harder (or easier) dependant on the present score. Here is another development that will make the game harder as the score goes up. You can choose the changing point for your game. This version will introduce a feature to modify the target (or Object)’s speed as the score increases!

Movehorizontal & Movevertical for each image control the direction and speed that the object moves. Earlier versions have made changes to these. For example find this code inside Sub DisplayScore()

```
If Score > 4 Then
  Movehorizontal(1) = 3 + Score : Movevertical(1) = Score - 2
Else
  Movehorizontal(1) = 0 : Movevertical(1) = 0
```

This forces Image1 to move once the Score is greater than 4. The 1 inside the brackets has been set up by the program and NSBasic to refer to Image1. You may have noticed that when the score changes above 4 Image 1 heads off in a downward and right direction. That is because the values are greater than 0. Values less than 0 like “-1” and “-5” will make it move up and left. The problem is how to get it to change speed relative to the Score. One way might be to use the present value and for a random number as well as the Score to all be included!

Try this revised code…

```
' If Score > 4 Then
   Movehorizontal(1) = Movehorizontal(1) - Score + (2*Score*Rnd())
   Movevertical(1) = 0
' Else
'   Movehorizontal(1) = 0 : Movevertical(1) = 0
' End If
```

The use of “Remming” out the condition means we can test the new code to see what it does. You might want to extend the game length too so that you can test the effect of it for longer. Do test it. Display the line below (on your home folder copy) when you think you know what is happening…

[redacted]

Take the remming out once you are happy.

Version 3.29

Now you can select many Scores and many objects for the game to get progressively harder.

We could use a Case Statement for Further variations… The code below works with Image2

Firstly we remove the if decision and replace with the Case code…

Previous Code	Alternative Code to use now…

`If Score > 1 Then`
`  Image2.hidden = False`
`Else`
`  Image2.hidden = True`
`End If`
	`Select Case Score`
	`  Case 0,1`
	`  Image2.hidden = True`
	`  Case Else`
	`  Image2.hidden = False`
	`End Select`

Test to see Image2 appears once the score reaches 2 or higher.

Now suppose we want to have a set speed if the Score is a value from 2 to 6. We must make sure it is displayed and then change the speed….	A further development when the score reaches 7 – 12…

`Select Case Score`
`  Case 0,1`
`  Image2.hidden = True`
`  Case 2,3,4,5,6`
`  Image2.hidden = False`
`  Movehorizontal(2) = 1`
`  Movevertical(2) = 0`
`  Case Else`
`  Image2.hidden = False`
`End Select`
	`Select Case Score`
	`  Case 0,1`
	`  Image2.hidden = True`
	`  Case 2,3,4,5,6`
	`  Image2.hidden = False`
	`  Movehorizontal(2) = 1`
	`  Movevertical(2) = 0`
	`  Case 7,8,9,10,11,12`
	`  Image2.hidden = False`
	`  Movehorizontal(2) = 2 + (Rnd()*4)`
	`  Movevertical(2) = 1`
	`  Case Else`
	`  Image2.hidden = False`
	`End Select`

It should now start to move…	The left right movement is more variable. In addition it will start to move.

Choose your rules for all Images!

Version 3.30

The Start box ought to size to the size of the border.

To do this we need to change how the Start box size is calculated. Find the sub….

```
Sub Button1Display()
  $(Button1).css("font-size", "50px");
  Button1.resize(20,20,400,400)
  Button1.value = "Start"
End Sub
```

Change the Button1resize line to… (You can copy from your home folder version).

Button1.resize(Border + 3 ,Border + 3, Form1.Width - (3+(Border * 2)), Form1.Height - (3+(Border * 2)))

Version 3.31 onwards

Add more images and extend the game!

Adjust the rules to suit you.