

**TegoSoft Inc.**

Copyright © 1994-1996 TegoSoft Inc. ® All Rights Reserved

*P.O.Box 389, Bellmore, NY 11710*

Web Site: <http://www.tegosoft.com>

 e-mail: [tegosoft@msn.com](mailto:tegosoft@msn.com)

# Lesson 29 - Refreshing the Screen


In this lesson you'll learn to write programs that take advantage of the **Refresh** method.


## Creating the Directory of the Project and Saving the Files of the Project


As usual, you'll start by creating the directory of the project, and saving the files of the project.

 Create the **C:\VBMyProg\Lesson29** directory.


You'll save the files of this lesson to this directory.

 Select **New Project** from the **File** menu of Visual Basic.

 Make the window of Form1 the selected window, select **Save File As** from the **File** menu of Visual Basic, and save the file as **MyCount.FRM** inside the **C:\VBMyProg\Lesson29** directory.

 Select **Save Project As** from the **File** menu of Visual Basic, and save the project as **MyCount.VBP** inside the **C:\VBMyProg\Lesson29** directory.

# Always Use Option Explicit

 Inside the general declarations section of Form1 type the following code:

```
' Force variable declarations  
Option Explicit
```

The **Option Explicit** statement forces you to declare variables before using the variables.

## Designing the Window of the MyCount Program

You'll design the window of the MyCount program so that it will look as shown in Figure 29.1.

 Set the properties of Form1 as follows:

**Name:** Form1

**Caption:** The MyCount Program

**BackColor:** White

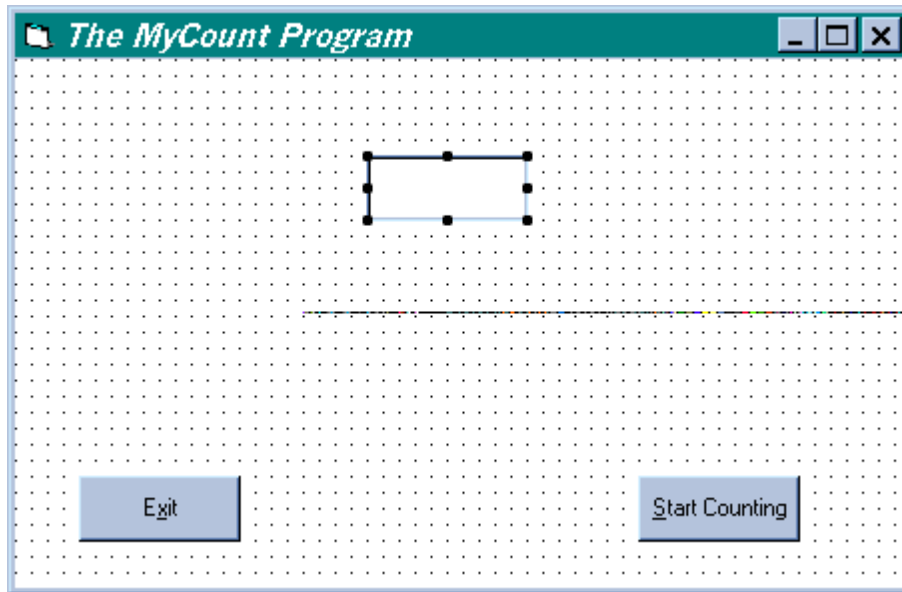



Figure 29.1. Form1 of the MyCount program

## Implementing the Exit Button

You'll now implement the Exit button:

 Place a CommandButton inside Form1, then set the properties of the CommandButton as follows:

**Name:** cmdExit

**Caption:** E&xit

## Attaching Code to the Click Event of the Exit Button

You'll now attach code to the **Click** event of the Exit button:

 Add the following code inside the **cmdExit\_Click()** procedure:

```
Private Sub cmdExit_Click()
```

=====


End

**End Sub**

So whenever the user clicks the Exit button, the MyCount program terminates itself.

## Placing a Label Control

You'll now place a Label control inside Form1.

 Place a Label control inside Form1. Then set the properties of the Label as follows:

**Name:** lblCounter

**Caption:** Make it empty

**Alignment:** 2-Cenetr

**BorderStyle:** 1-Fixed Single

## Placing the Start Counting Button

You'll now place a CommandButton inside Form1.

 Place a CommandButton inside Form1. Then set the properties of the CommandButton as follows:


**Name:** cmdStart

**Caption:** &Start Counting

## Counting

=====

You'll now write code that counts. The result of the counting will be displayed inside the **lblCounter** label:

 Type the following code inside the **cmdStart\_Click()** procedure:

```
Private Sub cmdStart_Click()  
  
Dim Counter  
For Counter = 1 To 1000  
  
    lblCounter.Caption = Str(Counter)  
  
Next  
  
End Sub
```

The code that you typed executes a **For()** loop that increases the variable **Counter** from **1** to **1000**. The **Caption** property of the **lblCounter** label displays the value of **Counter**.

 Select **Save Project** from the **File** menu of Visual Basic to save your work.

 Execute the MyCount program.


The MyCount program responds by displaying the number **1000** inside the **lblCounter** label. Depending on how fast is your PC, you may notice a small delay before the number **1000** is displayed.

## **NOTE**

As the **For()** loop is executing, the **Caption** property of the label is updated. So why don't you see the numbers **1,2, 3, ..., 1000** displayed inside the caption of the label? Because the PC is dedicating its resources

=====

for the execution of the **For()** loop. When the **cmdStart\_Click()** procedure terminates, the program updates the **Caption** label of the label. The last value that the **Caption** property has is **1000**, hence you'll see the **Caption** of the label updated with **1000** only after the **cmdStart\_Click()** procedure is terminated. In other words, in the current state the MyCount program, the window of the MyCount program is **not** being refreshed during the execution of the **For()** loop.

 Click the Exit button of the MyCount program to terminate the program.

## Refreshing the Screen

You'll now write code that refreshes the screen in each execution of the **For()** loop. This way, you'll see the label counting **1,2,3,...1000**.

 Modify the **cmdStart\_Click()** procedure so that it will look as follows:

```
Private Sub cmdStart_Click()  
  
Dim Counter  
  
For Counter = 1 To 1000  
  
    lblCounter.Caption = Str(Counter)  
  
    Form1.Refresh  
  
Next  
  
End Sub
```

You added the following statement:

=====

Form1.Refresh


The **Refresh** method refreshes Form1. This means that Form1 will re-display itself in each execution of the **For()** loop. Let's see this in action:

 Select **Save Project** from the **File** menu of Visual Basic to save your work.

 Execute the MyCount program.

 Click the **Start Counting** button.

The MyCount program counts, and now you see that the **Caption** of the label being updated with the numbers **1,2,3,...1000**.

 Experiment with the MyCount program. And then click its Exit button to terminate the program.

## What You Accomplished in This Lesson



You completed Lesson 29 of the Self Study Visual Basic tutorial. In this lesson you learned to take advantage of the **Refresh** method.

## Frequently Asked Questions



**Q1.** Should I execute the **Refresh** method whenever I change the value of the **Caption** property of a Label control?

**A1.** No, do not use the **Refresh** method every time you change the value of the **Caption** property of a Label control. The **Refresh** method refreshes

=====

the form. This means that if a control such as a Label control has a new value in its **Caption** property, the **Refresh** method will cause the immediate refresh of the **Caption** property of the Label. However, if the procedure that changes the **Caption** property of the Label is due to terminate soon, then there is **no** reason to execute the **Refresh** method, because when the procedure is terminated, the Form will refresh itself automatically. The only time you may want to execute the **Refresh** method is in situations such as the situation in the MyCount program. In the MyCount program, the program spends a lot of time inside the **For()** loop, and during the execution of the **For()** loop, the user does **not** see the **Caption** of the label being updated. Of course, the MyCount program is a simple program that just serves as means of illustrating the **Refresh** method. But you may encounter practical situations where your program will spend a lot of time inside a loop (e.g., when a procedure is involved in a very complex mathematical calculation), and in such cases, the refreshing mechanism is a useful mechanism to tell the user that "things are happening". Thus, the user will not think that the program is hanging, because some visual indication that the program is functioning will be displayed by utilizing the **Refresh** method.

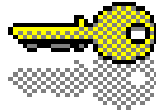
## Exam



1. Inside the **cmdStart\_Click()** procedure you have a **For()** loop that performs the counting. What other code can you use to accomplish the same thing that the **For()** loop accomplishes?

## Answers to Exam

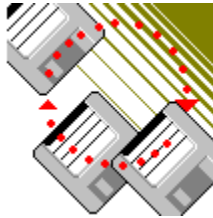
=====




1. The following **Do While** loop does the same exact job that the **For()** loop does:

```
Counter = 1
Do While Counter <= 1000
    lblCounter.Caption = Str(Counter)
    Form1.Refresh
    Counter = Counter + 1
Loop
```

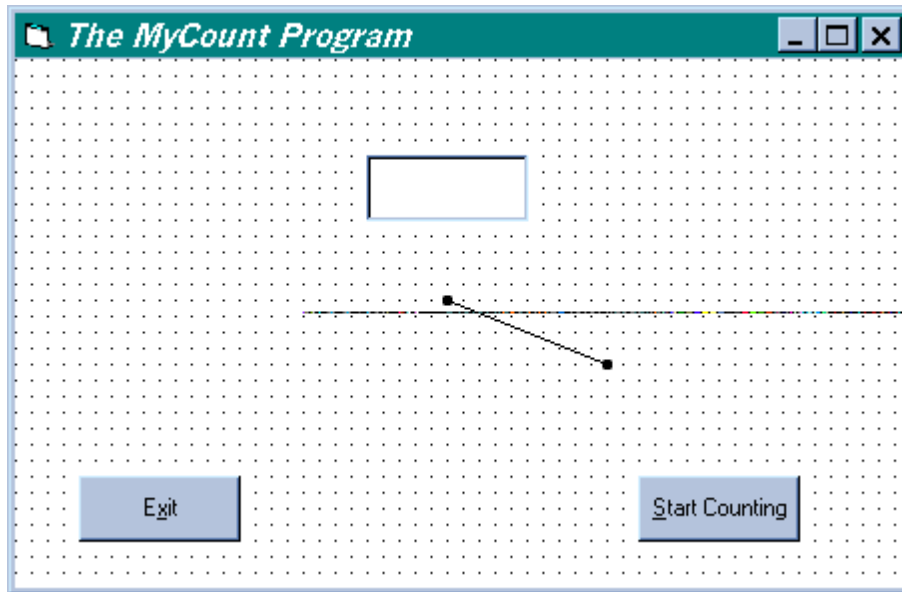
## Project



In this lesson you saw that the **Refresh** method can be used for an immediate refresh of the Label control. If the Form has other controls in it that have to be refreshed, then the **Refresh** method refreshes these controls as well.

 Place a **Line** control inside Form1, and make sure that the **Name** property of the Line control is **Line1**.

Your Form1 should now look as shown in Figure 29.2.



**Figure 29.2.** *Placing the Line control inside Form1.*

 Modify the **cmdStart\_Click()** procedure so that it will look as follows:

```
Private Sub cmdStart_Click()

Dim Counter
For Counter = 1 To 1000

    lblCounter.Caption = Str(Counter)

    Line1.X2 = Form1.ScaleWidth * Rnd()

    Line1.Y2 = Form1.ScaleHeight * Rnd()

    '''Form1.Refresh

Next
```

End Sub

You assign random values to the **X2** and **Y2** properties of the **Line** control.  
Note that you also commented out the **Refresh** statement:


```
'''Form1.Refresh
```

 Execute the MyCount program.

 Click the **Start Counting** button.

Because you commented out the **Refresh** statement, you will **not** see the caption of the Label being updated while the program is inside the **For()** loop, and you will **not** see the Line control move during the execution of the **For()** loop.


 Click the Exit button to terminate the MyCount program.

 Un-comment the **Refresh** statement inside the **cmdStart\_Click()** procedure.

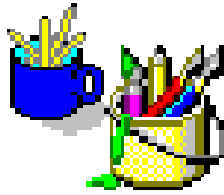
 Execute the MyCount program.

 Click the **Start Counting** button.

Because you now execute the **Refresh** statement inside the **For()** loop, you do see the caption of the Label being updated while the program inside the **For()** loop, and you do see the Line control move during the execution of the **For()** loop.

 Click the Exit button to terminate the MyCount program.

# Cosmetic Considerations



When implementing counters and other visual indication mechanisms that tell the user that the program is working inside a loop, you have to be careful not to display "moving things" too fast! If things are moving too fast, your user will think that there is something wrong with the program. For example, in the MyCount program, the **Caption** property of the label is changing so fast, that the user may think that there is something wrong with the program. Also, because the counter is changing so rapidly, the user can't read the value of the counter. **More important, the Refresh mechanism slows down the performances of the loop.** Without the **Refresh** statement, the program completes the execution of the **For()** loop much faster.

The solution is not to refresh the window in each execution of the **For()** loop. Rather, you can modify the code inside the **cmdStart\_Click()** procedure so that it will look as follows:

```
Private Sub cmdStart_Click()  
  
Dim LabelCounter  
Dim Counter  
For Counter = 1 To 1000  
  
    LabelCounter = LabelCounter + 1  
    If LabelCounter = 100 Then  
        lblCounter.Caption = Str(Counter)  
        LabelCounter = 0  
        Form1.Refresh  
    End If  
  
    Line1.X2 = Form1.ScaleWidth * Rnd()  
  
=====
```

```

        Line1.Y2 = Form1.ScaleHeight * Rnd()

        '''Form1.Refresh

Next

End Sub

```

You added another local variable:

```
Dim LabelCounter
```

Inside the **For()** loop you added the following code:

```

LabelCounter = LabelCounter + 1
If LabelCounter = 100 Then
    lblCounter.Caption = Str(Counter)
    LabelCounter = 0
    Form1.Refresh
End If


```

The receding code increases **LabelCounter** by 1, and then an **If** statement is executed to check the value of **LabelCounter**. If **LabelCounter** is not equal to **100**, the code under the **If** statement is **not** executed. Because the **Refresh** method now appears under the **If** statement (and you commented out the original **Refresh** statement), the Form will **not** be refreshed for each execution of the **For()** loop. Rather, the code under the **If** statement is executed every **100** counts (and therefore the **Refresh** statement is executed once in every **100** counts of the **For()** loop). So now the **For()** loop will be executed much faster, because the **Refresh** method is executed only 10 times during the entire execution of the **For()** loop.

 Execute the MyCount program.

 Click the **Start Counting** button.

Note that the entire **For()** loop is executed much faster, because the **Refresh** method does not impede the performances. Of course, you do not see the caption of the label changes for each count of the **For()** loop. Rather, you see the caption of the label changes every **100** counts of the **For()** loop.

 Click the Exit button to terminate the MyCount program.

## How To Contact TegoSoft



You can contact TegoSoft Inc. by any one of the following methods:

- Use TegoSoft Internet Web site:

<http://www.tegosoft.com>

- Send TegoSoft an e-mail:

[tegosoft@msn.com](mailto:tegosoft@msn.com)

- Send TegoSoft a letter:

*TegoSoft Inc.*

*P.O.Box 389*

*Bellmore, NY 11710*


*USA*

## Technical Support



If you have a technical question, you can post the question to the TegoSoft Technical Support staff, and they will try to answer your question.

The **best** way to post a technical question is by sending TegoSoft an e-mail.

 The e-mail of TegoSoft is:  
tegosoft@msn.com

**When sending TegoSoft an e-mail with a technical question, please follow the following format:**

Date: \_\_\_\_\_  
Your name: \_\_\_\_\_  
Company (if applicable): \_\_\_\_\_  
Your phone number: \_\_\_\_\_  
Your e-mail: \_\_\_\_\_  
Country (if not USA): \_\_\_\_\_  
State (if inside USA): \_\_\_\_\_

**Operating System used:** \_\_\_\_\_  
**Programming language and version :** \_\_\_\_\_

**My technical question is:**

## **Copyright © and Notices**

Copyright © 1994, 1995, 1996 by TegoSoft Inc. ® All Rights Reserved

### **TegoSoft Self Study Tutorials & Software**

**Copyright ©1994, 1995, 1996 by TegoSoft Inc. ® All Rights Reserved**

Although precaution was taken in the preparation of this document, TegoSoft assumes no responsibility for errors or omissions. TegoSoft is

=====

**TegoSoft Visual Basic 4 Sel Study Tutorial - Lesson 29**

not liable for damages resulting from the use of the information contained in this document. Use this document at your own risk. This document is copyright protected. This means that you should treat this document like any other copyright material. No part of this document should be copied in any way. You are not allowed to use this document or any part of this document for any purpose other than read it. You are not allowed to publish this document or any part of this document in any way. You are not allowed to sell this document or any part of this document, you are not allowed to incorporate this document or any part of this document in any book, magazine, Web Site, Electronic forums, disks, CDs, or any other media. The only thing that you are allowed to do with this document is read it for the sole purpose of studying the material that is presented in this document. If this document includes software, the software must be treated in the exact same way that this document is treated. You are not allowed to distribute the software in any way. The sole purpose of supplying the accompanying software is to enable you to experience with it. No part of this document should be modified. If accompanying software is included with this document, you are not allowed to modify the software.

Rev. 031796-1