

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


Lesson 12 - Declaring Variables


In this lesson you'll learn about declaring variables in Visual Basic and about the importance of the **Option Explicit** statement.


Create the Directory of the MyBasic Program, Start Visual Basic 4, and Save the Form and the Project Files

In this lesson you are going to create several files. So first of all, let's create a directory where the files that you'll create during the course of this lesson are saved.


 Create the **C:\VBMyProg\Lesson12** directory on your hard drive,

 Start Visual Basic, and then select **New Project** from the **File** menu of Visual Basic.

 Select **Project** from the **View** menu of Visual Basic to display the Project window. Click the **View Form** button of the Project window to make **Form1** the selected form.


 While Form1 is the selected window, select **Save File As** from the **File** menu of Visual Basic.

Visual Basic responds by displaying the **Save File As** dialog box.

 Save the Form file as **MyBasic.frm** inside the **C:\VBMyProg\Lesson12** directory.


 Select **Save Project As** from the **File** menu of Visual Basic.

Visual Basic responds by displaying the **Save Project As** dialog box.

 Save the project file as **MyBasic.vbp** inside the **C:\VBMyProg\Lesson12** directory.

Set the Properties of Form1

You'll now set the properties of **Form1**.

 Make the **Form1** window the selected window, and then press the F4 key on your keyboard to display the **Properties** window of Form1.


 Set the properties of **Form1** as follows:

Property	Setting
Name	Form1
Caption	The MyBasic Program
BackColor	White

The Option Explicit Statement

One of the most important statements in Visual Basic is the **Option Explicit** statement.

To appreciate the **Option Explicit** statement, perform the following experiment:

 Double click inside Form1 to display the Code window, set the **Object** list box to **(General)**, and set the **Proc** list box to **(declarations)**. The Code window should now look as shown in Figure 12.1. The section of the Code window that has the Object list box set to **(General)** and the

Proc list box set to **(declarations)** is called the general declarations section.

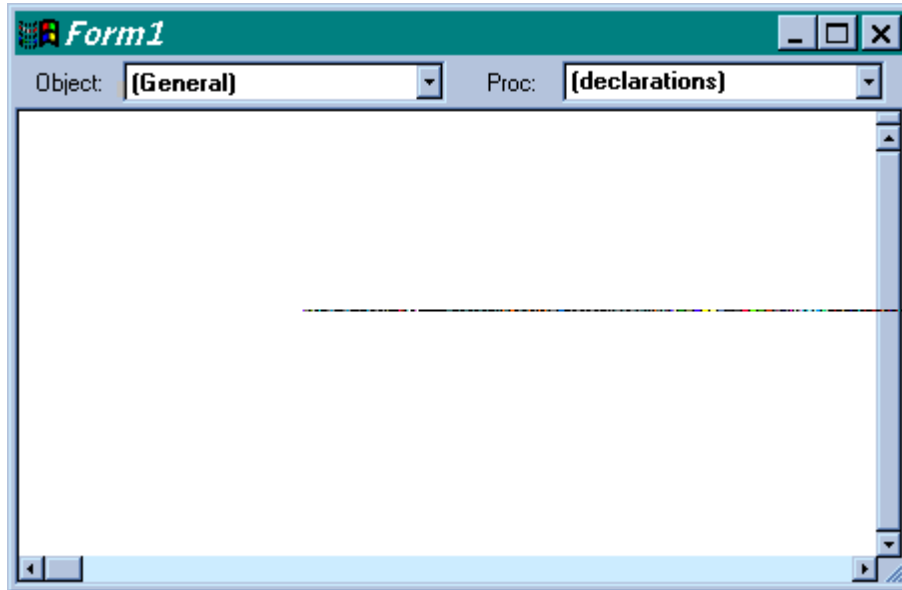




Figure 12.1. *The General declarations section of the Code window.*

 Depending on the current settings of your Visual Basic, Visual Basic may already insert the statement **Option Explicit** inside the general declarations section as follows:

Option Explicit


 If you see the Option Explicit statement, then comment it out as follows:

```
' Option Explicit
```

The reason you were instructed to comment out the **Option Explicit** statement is because you will now see the effect of not having the Option Explicit inside the general declarations section.


Adding the Test Option Explicit Button

You'll now add a CommandButton inside Form1.

 Place a CommandButton inside Form1, and then set its properties as follows:

Name: cmdTestOptionExplicit

Caption: Test Option Explicit

 Make the button wider, so that the entire caption of the button will fit on a single line, and move the button to the lower part of the form.

Your Form1 should now look as shown in Figure 12.2.

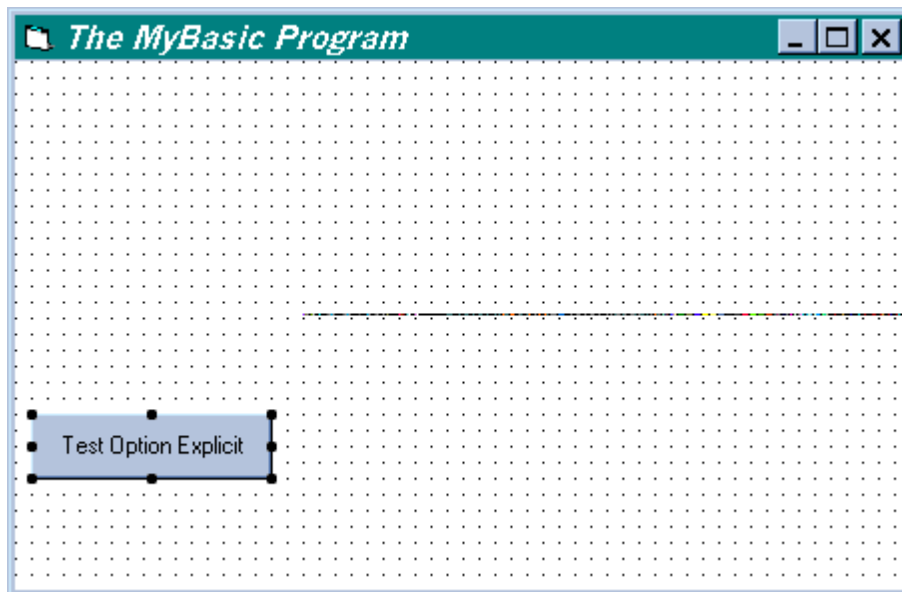



Figure 12.2. Form1 with the Test Option Explicit button in it.

Attaching Code to the Click Event of the Test Option Explicit Button

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

 Double click the cmdTestOptionExplicit button to display the Code window, set the Object list box to cmdTestOptionExplicit, and set the Proc list box to Click.

Visual Basic responds by displaying the cmdTestOptionExplicit_Click() procedure as follows:

```
Private Sub cmdTestOptionExplicit_Click()
```

```
End Sub
```

 Add code inside the cmdTestOptionExplicit_Click() procedure as follows:

```
Private Sub cmdTestOptionExplicit_Click()
```

```
MyVariable = 3  
HerVariable = 2  
OurVariable = MyVariable + HerVariable  
Print "OurVariable=" + Str(OurVariable)
```

```
End Sub
```

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

The code that you typed inside the cmdTestOptionExplicit_Click() procedure sets the value of the MyVariable variable to 3:

=====

```
MyVariable = 3
```

Then you set a variable called HerVariable to 2:

```
HerVariable = 2
```

Then you set the value of a variable called OurVariable as follows:

```
OurVariable = MyVariable + HerVariable
```


Because MyVariable is equal to 3 and HerVariable is equal to 2, you expect the variable OurVariable to be equal to 5 (2+3=5).

The last statement that you typed inside the cmdTestOptionExplicit_Click() procedure is as follows:

```
Print "OurVariable=" + Str(OurVariable)
```

The preceding statement prints the value of the OurVariable variable.

Let's see your code in action:

 Execute the MyBasic program, and then click the Test Option Explicit button.

The MyBasic program responds by displaying the text:

OurVariable=5

as shown in Figure 12.3. (Look at the upper left corner of the window), the **Print** statement that you executed displays the string on the first line of the window.

=====

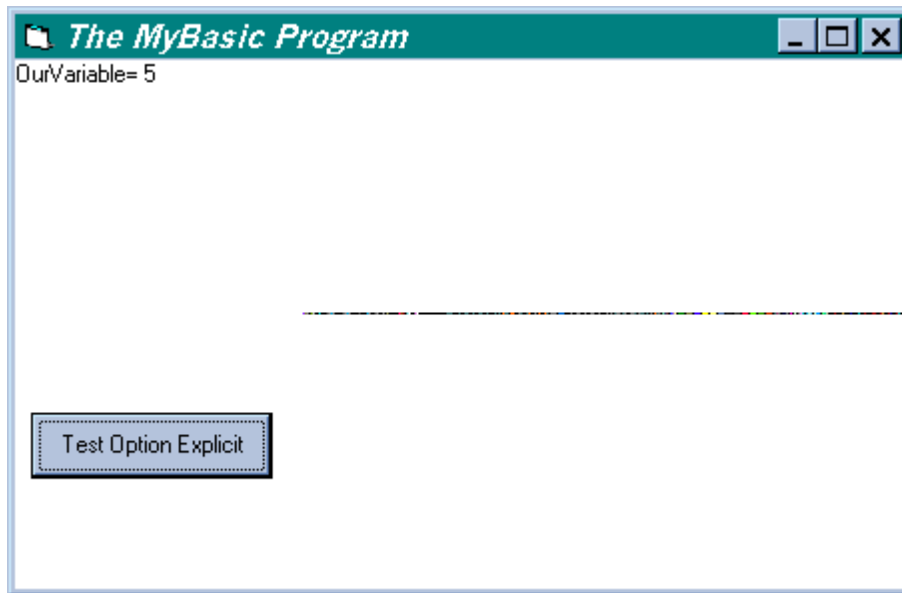



Figure 12.3. *The value of the OurVariable variable is displayed on the upper left corner of the window.*

So if you typed the code exactly as instructed, you see that the MyBasic program working as expected.

 Terminate the MyBasic program.

Making an Error On Purpose...

Now you'll modify the MyBasic program. In particular, you'll make an error on purpose, and see the effect of this error on the result.

 Modify the code inside the cmdTestOptionExplicit_Click() procedure as follows:

```
Private Sub cmdTestOptionExplicit_Click()
```

```
    ' MyVariable = 3
```

```
    MyVariable = 3
```

```
=====
```

```
HerVariable = 2
OurVariable = MyVariable + HerVariable
Print "OurVariable=" + Str(OurVariable)
```

End Sub


You commented out the first statement:

```
' MyVariable = 3
```

And you added the following statement:

```
MyVariable = 3
```

The preceding statement sets the value of **MyVariable** to 3. Yes, sometimes you make a typing mistakes, and in this case you did not type the **i** after the **r** in the **MyVariable** variable.

 Execute the MyBasic program. And then click the Test Option Explicit button.

Visual Basic responds by displaying the following text on the upper left corner of the window:

```
OurVariable=2
```

 Terminate the MyBasic program.

To understand why Visual Basic thinks that OurVariable is equal to 2, look at the statement that calculates OurVariable:

```
OurVariable = MyVariable + HerVariable
```


=====

HerVariable is equal to 2. And because you made a typing mistake, the value of **MyVariable** was never set to 3. (The variable MyVariable is set to 3, but you did not set the value of **MyVariable**). Because MyVariable was not initialized to 3, Visual Basic uses the value **0** for the MyVariable variable. So putting it all together, OurVariable is equal to 0+3 which is 3.

Detecting Typos

It would be nice if Visual Basic automatically notice that you made a typing errors. In the MyBasic program, it is relatively a simple matter to discover the typing error. But if your program has some very complex calculations that include many variables, it will be very difficult to discover the typing error.

Luckily, Visual Basic lets you use the Option Explicit statement, a statement that basically prevents you from making silly typo errors. To see the Option Explicit statement in action, perform the following experiment:

 Double click inside the Form1 to display the Code window, set the Object list box to **(General)**, and set the Proc list box to **(declarations)**. Then type the following statement inside the general declarations section:

Option Explicit

Your general declarations section should now look as shown in Figure 12.4.

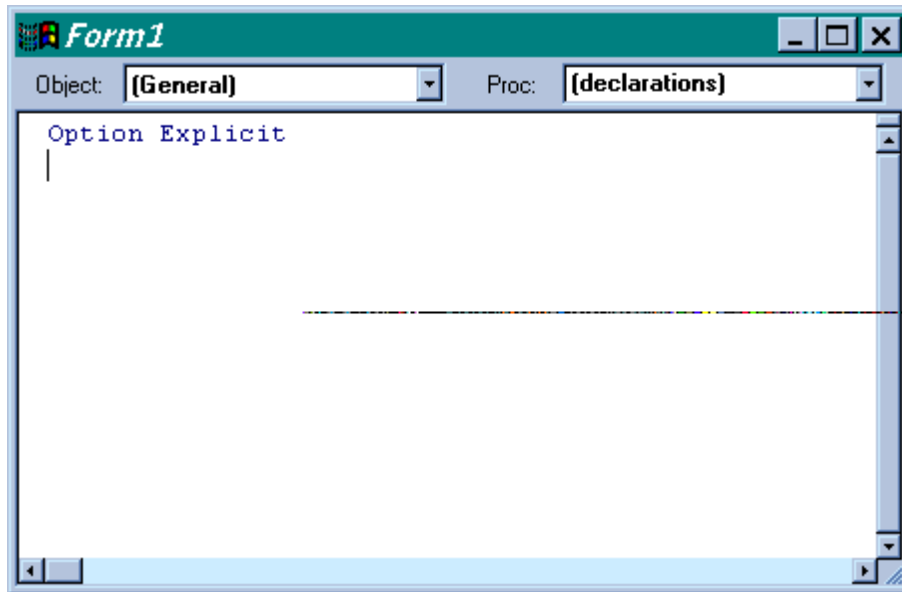



Figure 12.4. *Typing the Option Explicit statement inside the general declarations section.*

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

 Execute the MyBasic program.

 Click the Test Option Explicit button.

Visual Basic responds by displaying an error dialog box. The dialog box tells you that a variable is not defined.

 Close the error dialog box, and then terminate the MyBasic program.

You included the Option Explicit statement inside the general declarations section. This means that Visual Basic expects you to declare all the variables that your program uses. If you use a variable without first declaring it, you'll get an error.

Declaring the Variables

Modify the code inside the cmdTestOptionExplicit_Click() procedure so that it will look as follows:

Private Sub cmdTestOptionExplicit_Click()

```
Dim MyVariable
Dim HerVariable
Dim OurVariable

MyVariable = 3
'MyVariable = 3
HerVariable = 2
OurVariable = MyVariable + HerVariable
Print "OurVariable=" + Str(OurVariable)
```

End Sub

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

You added the following statements:

```
Dim MyVariable
Dim HerVariable
Dim OurVariable
```


The preceding statements tell Visual Basic that your procedure has permission to use the variables MyVariable, HerVariable and OurVariable.

Note that you un-commented the statement that sets the value of MyVariable to 3, and you commented out the typing error:

```
MyVariable = 3
```



```
'MyVariable = 3
```

 Execute the MyBasic program, click the Test Option Explicit button, and verify that the program reports that OurVariable is equal to 5.

 Terminate the MyBasic program.

Making a Typing Error Again...

Now you'll return the typing error:

 Modify the cmdTestOptionExplicit_Click() procedure so that it will look as follows:

Private Sub cmdTestOptionExplicit_Click()

```
Dim MyVariable
Dim HerVariable
Dim OurVariable

' MyVariable = 3
MyVariable = 3
HerVariable = 2
OurVariable = MyVariable + HerVariable
Print "OurVariable=" + Str(OurVariable)
```

End Sub

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

You again commented out the "good statement", and you un-comment the typing error:

=====

```
' MyVariable = 3
MyVariable = 3
```

 Execute the MyBasic program.

Good News! Visual Basic stops the execution, and a dialog box informs you that the **MyVariable** is not defined!

 Terminate the MyBasic program.

Well, because the **MyVariable** is **not** declared with the **Dim** statement, Visual Basic refuses to continue the execution.

To summarize, because you included the Option Explicit statement inside the general declarations section, Visual Basic expects you to declare variables before you use the variables. If you made a typing error (that is, if you type a variable different than the way it is typed in its **Dim** statement), Visual Basic will stop the execution, and you'll receive an error dialog box.

The Dim Statement

You saw that the **Dim** word is used for declaring a variable. For example, to declare the MyVariable variable, you use the following statement:

```
Dim MyVariable
```

In the preceding statement, you tell Visual Basic to declare the MyVariable variable, but you did not tell Visual Basic whether the MyVariable is an integer, string, date, or other type of variable.

Well, it is ok to declare variable without specifically specifying the type of the variable. Nevertheless, if you want to declare a variable in a more specific way, you can do so. For example, to declare the variable MyVariable as an integer, you can use the following statement:

```
=====
```

```
Dim MyVariable as Integer
```

Likewise, if you want to declare the variable HisMessage as a string, use the following statement:

```
Dim HisMessage as String
```

What You Accomplished in This Lesson



You completed Lesson 12 of the Self Study Visual Basic tutorial. In this lesson you learned about the importance of the **Option Explicit** statement, and how to use the **Dim** statement for declaring variables.

Frequently Asked Questions



Q1. I rarely make typing error. Should I use the **Option Explicit** statement?

A1. Yes, use the **Option Explicit** statement. Everybody makes typing errors from time to time.

Q2. Why would I want to declare the type of the variable? That is, it is easier to use the statement:

```
Dim MyVariable
```

than to declare the variable as follows:

=====

```
Dim MyVariable as Integer
```

A2. The advantage of not specifying the type of the variable is that you have less to type.

There are two disadvantages of not specifying the type of variable:

If you do not specify the type of variable, Visual Basic assumes that the variable can be of any type. This means that the memory storage that Visual Basic reserves for the variable is larger than the memory needed for the variable. For example, suppose that you are declaring a variable as an integer. Depending on the operating system that you are using, an integer may require 4 bytes of memory. Yet, a variable of type Long (a variable that is capable of holding very large numbers) may require double the amount of bytes that are needed for storing an integer. If you do not specify the type of variable, Visual Basic assumes that the variable can be of type that requires more bytes that are really needed for this variable.

The other disadvantage of not specifying the type of variable is that when the variable occupies more bytes, the time it takes to perform calculations is longer. For example, it takes longer time to add two Long numbers than the time it takes to add two Integer numbers.

Typically however, unless your program uses a huge amount of variables, the use of additional memory due to the fact that you did not specify the type of variable is not a problem. And also, if your program is not involved in many calculations, the difference in performances is negligible.

Exam

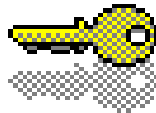


1. The Following statement is valid:

```
Dim A, B, C
```

- (a) True
- (b) False

Answers to Exam



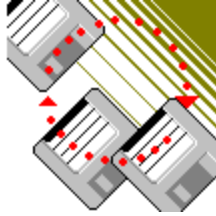
1. (a) True, you can include more than one variable on the same Dim line. However, all the variables that are declared with the same Dim statement must be of the same type. So the following statement are all valid:

```
Dim A, B, C
```


```
Dim MyString, HerString, HisString as String
```

```
Dim MyNumber, HerNumber, HisNumber as Integer
```


Project



The **Print** statement that you used prints inside the window of the program (see Figure 12.3). Modify the program so that the font used is larger.

 Make Form1 the selected form, then press F4 to display the Properties window of Form1.

 Set the Font property of Form1 to Size 14.

 Execute the program, click the Test Option Explicit button, and note that now the font used is larger (see Figure 12.5).

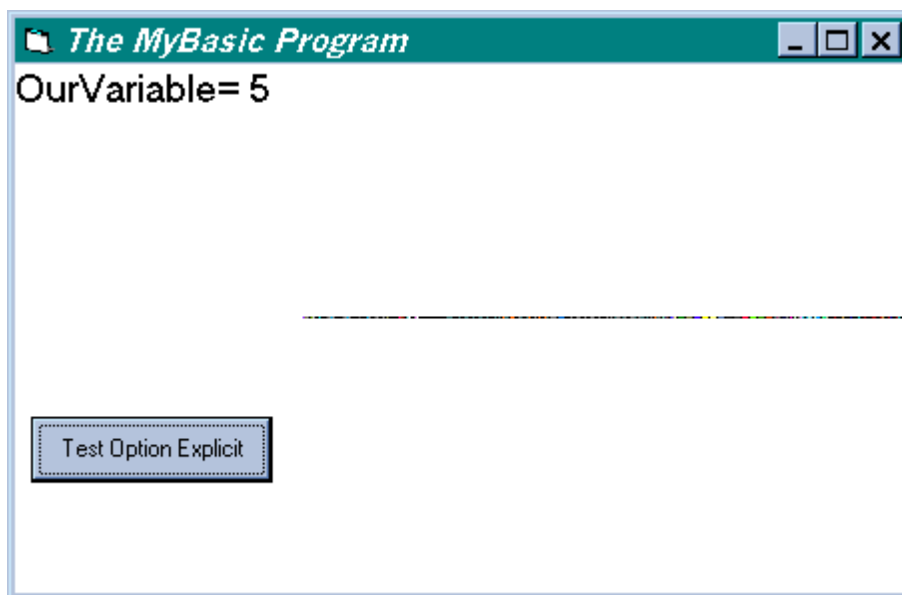
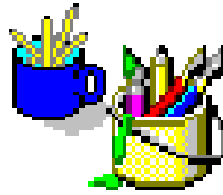


Figure 12.5. *Using a larger font.*

 Terminate the MyBasic program.

Cosmetic Considerations



Typically, when implementing programs such as the MyBasic program, you do not make any efforts to make the program pretty. Occasionally, you will want to test something, and you will want to implement a test program that serves as a test program to test a feature of Visual Basic. In such a case, make the program as simple as possible, and do not pay attention to the cosmetic aspect of the 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

- 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

=====

TegoSoft Visual Basic 4 Self Study Tutorial - Lesson 12

Page 19 of 20

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 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