Module 8

Embedded Basic

Materials Required for Module

r *Microsoft Access Basic: An Introduction to Programming*

ModuleObjectives

Lesson 1 - Microsoft Access Basic and the Immediate Windows

Upon completion of this lesson you will be able to:

r Define the terms: MS Access Basic, function, and sub.

r Create a new module and save it as a text file.

Lesson 2 - Writing a New Function

Upon completion of this lesson, you will be able to:

r Create and compile a function which accepts arguments.

r Add comments to a function.

Lesson 3 - Microsoft Access Basic Fundamentals

Upon completion of this lesson, you will be able to:

r Set variables.

r Create decision and loop structures.

Lesson 4 - Debugging Your Access Basic Code

Upon completion of this lesson, you will be able to:

r Define compilation, run-time, and logic errors.

r Single step through a procedure.

r Run code in the Immediate Window.

Lesson 1

Microsoft Access Basic and the Module Window

Reading Assignment:

r *Microsoft Access Basic*
Chapter 1: *Microsoft Access Basic and the Module Window.*

As You Read

r If you are not an experienced programmer the idea of learning to write Access Basic code may seem intimidating at first. However, Access offers a lot of help. There is example code in On-Line Help, a programmer's reference which documents each function, and, best of all, a manual designed to teach you the basics. As you work through the reading assignments try each example. The more you code the simpler it will seem.

Exercises

1) What is a procedure?

   See page 2 - A procedure contains a series of MS Access Basic statements that perform some operation or calculate a value.

2) What is the difference between a function and a sub?

   See page 2 - A procedure that accepts zero or more arguments and returns a value. A sub cannot return a value.

3) What hot-key lists all modules and procedures in the database?

   See page 4 - F2

4) Can more than one person work on a module at the same time and save changes to it without overwriting each other's code?

   See page 4 - One person must save their changes to a text file.

5) How do you save a module as a text file?

   See page 5 - Choose Save Text from the File menu.

6) How do you import a module that you have saved out as a text file?
   See page 5- Choose File Load Text

Try This

1) Create a new module.

2) Place your cursor in the string Option Compare Text.

3) Press F1 for Help.

4) What is the purpose of the string Option Compare Text?

   Specifies whether text comparison for the module is Binary, Text or Database for the modules. (Binary=case-sensitive; Text=insensitive)

5) If you leave this statement out what is the default for Access?

   The default is Binary comparison of text strings.

6) Close Help and return to the new module.

Lesson 2

Writing a New Function

Reading Assignment:

r *Microsoft Access Basic*
Chapter 2: *Writing a New Function*

As You Read

You can create custom functions and use them just like you can use the functions provided by Access. This allows you to quickly find information and can make reporting and data entry quicker. For example, a research group enters numeric test results in base 16 but wants to create reports which reflect the equivalent numbers in base 8. Instead of entering the conversion formula in each control of each report and form where that information will be needed, they might create an Embedded Basic function which they can call from anywhere in their database. What are some custom functions you might use?

Hint: Help contains a reference of the Access functions

Exercises

Answer These

1) What is the limit on the length of a function name?

   See page 2 - 40 characters

2 Can you have spaces in function, argument, variable, or constant names in Access?

   See page 3 - No

3) What do you assign the results of a function to in order to have it return a value?

   See page 3 - The name of the function.

5) What is wrong with the following line of code:

   DueDate = DateSerial 'That's an Access function' (Year(anyDate), 1, 1)
   page 5 - comments can only be at the end of a line of code or on a separate line.

Try This

1) Open a module.
2) Create the following function:

```
Function Sum()
        MsgBox "Hello"
End Function
```

3) In the Immediate Window enter "X=Sum()". What happened?
4) Create a new form. Add an unbound text box. Enter "=Sum()". What happened? Why?

Lesson 3

Microsoft Access Basic Fundamentals

Reading Assignment:

r *Microsoft Access Basic*
Chapter 3: *Microsoft Access Basic Essentials*

As You Read

r Does it matter what you call your variables? Imagine that you want to create three different Access Basic procedures that you will call from a form. The first will calculate the average stock price for the month, the second returns your total investment, and the third calculates the percentage of return on your investment. There are several variable names that you might want to use in all three of these procedures. Current_Date, Current_Price, Start_Date, End_Date, SubTotal, Next, etc. You could declare these as global variables, making them accessable to all procedures but what limit does Access place on the amount of global variables?

Could not find limits of globals.

If you were to make your modules available to your co-

workers it is very likely that they might already have used the variables Start_Date, End_Date, or SubTotal in their own procedures.

Exercises

1) How would you declare a variable called *Monthend*?

   See page 2 - Dim Monthend

2) Do you have to specify a data type when you declare variables in Access Basic?

   See page 2 - No, Access variables default to the Variant data type.

3) What types of data can the Variant data type store?

   See page 2 - numeric, date/time, and string data. And it can be empty or null.

4) What does the DoCmd statement do?

   See page 2 - The DoCmd statement executes a macro action.

5) What number specifies the object type for modules?

   See page 3 - 5

6) What are the commands that you cannot execute with the DoCmd statement? Next to each list the Access Basic Equivalent that you would use instead.

   page 4 -

AddMenu	*No Equivalent*
MsgBox	*MsgBox*
RunApp	*Shell*
RunCode	*procedure call*
SendKeys	*SendKeys*
SetValue	*simple assignment*

Stop AllMacros	*No Equivalent*
StopMacro	*No Equivalent*

7) Can you use the GoTo() function in a sub procedure?

   Help - No. See the Help topic Sub.

8) If you declare a Function as Private can you call it from a form or report?

   See page 5 - No

9) What is the value of Total after the following code is executed?

```
Total = 0
If Total Then Total = -1
Total = Total +1
If Total Then Total = Total - 2
```

See page 6: -1

Exercises

10) What is the value of Total after the following code is executed?

```
Total = 0
If Total Then
Total = 10
ElseIf Total < 1 Then
Total = 100
Else
Total = 1000
End If
```

See page 6 - True, Total is set to 100 and, since a condition has been met skip to the End If.

11) What message will be displayed when the following code is run?

```
Total = 7
Select Case Total
   Case 1,2
      MsgBox "Have a nice day"
   Case 3,4,5
      MsgBox "Good Morning"
   Case 6
      MsgBox "Good Bye"
   Case Else
      MsgBox "Good Evening"
    End Select
```

page 8 - "Good Evening"

12) What is the value of Total after the following code is executed?

```
Total = 10
' -------- a Do Loop
Do Until Total > 0
     Total = Total - 1
Loop
' --------  another Do Loop
Do
     Total = Total - 1
Loop Until Total > 0
```

See page 9 - Total is equal to 9. The first loop does not execute because the condition (Total > 0) is true. The second executes once.

Exercises

13) In the example on page 11, if there were 15 forms loaded when this function was executed how many times would the For Loop execute?

See page 11 - It would execute 15 times since it would be incremented from 0 to 14 (Forms.Count - 1 because it starts at 0).

14) What statement could be placed within the statement block following

If Forms(i).Name - ForName Then

that would cause the execution of this function to stop as soon as IsLoaded was set to True?

Page 11 - Exit For or Exit Function

Lesson 4

Debugging Your Access Basic Code

Reading Assignment

r *MS Access Basic*
Chapter 4: *Debugging Your Access Basic Code*

As You Read

r If you learn nothing else about Access Basic you will want to be able to use the debugging tools quickly and effectively and understand the different types of error messages and their causes. Customers don't often call just to share the elegance of a section of code they have written; they call when it breaks and they don't understand the steps they need to take in order to fix it.

Exercises

1) What is a compilation error?

   See page 1 - An error caused by a statement that is not constructed correctly. Compilation errors are detected when your code is compiled.

2) What is a run-time error?

   See page 1 - Run-time errors occur when a statements attempts the impossible.

3) What is a logic error?

   See page 1 - Errors in the design of your code which cause incorrect results.

4) What does the Options Explicit statement do?

   See page 2 - Causes MS Access to verify that each variable in your module has been explicitly declared.

5) What will cause Access Basic to stop execution?

   See page 3 - At a run-time error, at a breakpoint, or at a Stop statement.

6) What happens to suspended procedures when you choose Reinitialize from the Run menu?

   See page 4 - They are reset.

7) If your code is halted at a breakpoint, has the current statement,(the one outlined) been executed?

   See page 5 - No

8) Can you save breakpoint setting from one session of Access to the next?

   See page 5 - No, they are cleared when you close your database. Use Stop instead of Breakpoints

9) How do you start the execution of your code on a specific line?

   See page 7 - Place you cursor in that line of code and choose Set Next Statement from the Run menu.

Exercises

Try This

1) Open a new module
2) Create a global variable Total:

   "Dim Total"
3) Create a sub procedure call Add_Ten by typing Sub Add_Ten() in the main Module window.
4) Focus automatically changes to the Sub Procedure window for Add_Ten.
5) Declare a local variable *Temp_Total*, set it to 10 and then add it to Total. Include two Debug.Print statements so that you print the value of Total both before and after adding Temp_Total to it:

```
Sub Add_Ten ()
Dim Temp_Total
    Debug.Print "total is: "; Str(Total)
    Temp_Total = 10
    Total = Total + Temp_Total
    Debug.Print "total is: "; Str(Total)
End Sub
```

6) Compile your procedure to make sure you entered everything correctly.
7) Switch to the Immediate Window and run your procedure by typing *Add_Ten* and pressing Enter. What is the ending value of Total?

   10
8) Enter *Add_Ten* again. What is the value of Total?

   20
9) Choose Reinitialize from the Run Menu.
10) Enter *? Total*. What is the value of *Total* now?

    NULL

The PSS Challenge

The report Alphabetical List of Products prints the Left character of each group of product. To troubleshoot it create code which will print out the entire product name each time the leftmost character is calculated. The following example has two mistakes in it. What are they?

```
Sub Debug_Report ()
   Debug.Print [Product Name]
End Sub
```

The reference to [Product Name] will not work. You would also have to call this sub from a function. Make this a function instead. Using a MsgBox function might also be more useful.

When would you want to enter Options Explicit in the declarations section of your modules?

To force all variables to be declared. This will prevent duplication of names. This makes a program easier to read..

On-Your-Own Lab

Since many of the people in your Address Book database do not have any dependents you may not want to always see the Dependents sub-form. In this lab you will create a module which automatically hides the embedded Dependents sub-forms each time the user moves to a new record. You will also place a button on the New Record Entry form which will allow the user to display the Dependents sub-form if they need to enter a new Dependent or view existing one.

1) To verify that you can thoroughly test your MS Access Basic code verify that you have at least one record with no Dependents in your database.
2) Open the New Record Entry form in Edit mode. After you have found out what the Control Name of the embedded sub-form is, switch to Browse mode.
3) Start a new module and save it as "Record Entry Controls".
4) In the main module window enter "Sub Hide_SubForm" to start the first sub procedure.
5) Enter Forms![New Record Entry]!Embedded0.Visible = 0
6) In the Immediate Windows enter "Hide_SubForm" and verify that the sub-form is no longer visible in New Record Entry.
7) In the main module window enter "Sub Show_SubForm" to start the second procedure.
8) Enter Forms![New Record Entry]!Embedded0.Visible = 1 and execute the Show_SubForm procedure in the Immediate Window to verify that the sub-form is now visible.
9) You cannot execute sub procedures from Forms or Reports, only functions. Now you need to create functions which will start each of the procedures you just tested so that you can use them with your form. Create a new Hide() function and enter the statement Hide_SubForm. Then create a function Show() which executes the sub procedure Show_SubForm.
10) You now see five options when you drop down the procedure list: (declarations), Hide, Hide_SubForm, Show, and Show_SubForm.
11) Save your work, activate the New Record Entry form, and switch to Design Mode.
12) Add a command button to your form with the caption "Enter New Dependents" and the OnPush property "=Show()".
13) Click on the gray area to the right or below your form to display the Form properties. Type =Hide() for the property which is applied before each new record is displayed.

Instructor-Led Module Review

1) Answer Questions

Can more than one person work on a module at the same time and save changes to it without overwriting each other's code?

See page 4 - One person must save their changes to a text file.

How do you save a module as a text file?

See page 5 - Choose Save Text from the File menu.

2) Review As You Read and Points to Ponder

Try This

1) *Open a module.*
2) *Enter* Function Sum().
3) *Why did you receive an error message?*

3) Present one or two possible lab solutions.

4) Discuss various participant solutions