
VBA DEBUGGING TECHNIQUES - COMPLETE NOTES

1. USING BREAKPOINTS (F9)

Definition:

A breakpoint pauses code execution at a chosen line.

How to Use:

- Click on any code line
- Press F9 to set a red dot (breakpoint)
- Run code with F5
- Code stops at breakpoint
- Hover over variables to see their values

Example:

```
Sub Debug_Breakpoint()
    Dim a As Integer, b As Integer, c As Integer
    a = 10
    b = 5
    c = a * b    ' Press F9 on this line
    MsgBox "Result: " & c
End Sub
```

When you run this macro, execution pauses on the `c = a * b` line.
You can then hover over variables (a, b) to see their values.

2. STEP THROUGH CODE (F8)

Definition:

Execute your code line by line to watch values change.

How to Use:

- Place cursor at top of Sub
- Press F8 to run first line
- Press F8 again for next line
- Continue pressing F8
- Watch variables update in real time

Example:

```
Sub Debug_StepThrough()
    Dim x As Integer, y As Integer, z As Integer
    x = 8
    y = 4
    z = x / y    ' Step through with F8
    MsgBox "Answer: " & z
End Sub
```

Run this and press F8 repeatedly – each line executes step by step.
You'll see variables update in the Locals Window or when hovering.

3. IMMEDIATE WINDOW (Ctrl+G)

Definition:

Lets you print values or run code interactively.

How to Use:

- Press Ctrl+G to open Immediate Window
- Type `Debug.Print` to print values

- See results instantly
- Use for quick testing

Example:

```
Sub Debug_ImmediateWindow()
    Dim i As Integer
    For i = 1 To 5
        Debug.Print "i = "; i    ' Prints to Immediate Window
    Next i
End Sub
```

Type this in Immediate Window:
Debug.Print "Test" ' Shows: Test

4. WATCH WINDOW

Definition:

Monitor a variable's value automatically as code runs.

How to Use:

- Right-click on variable name
- Select "Add Watch"
- Watch Window appears
- Shows variable value changing
- Updates live during execution

Example:

```
Sub Debug_WatchWindow()
    Dim total As Integer, i As Integer
    total = 0
    For i = 1 To 3
        total = total + i    ' Add watch on "total"
    Next i
    MsgBox "Final total: " & total
End Sub
```

Right-click on "total" → Add Watch
VBA will track its value as the loop runs.
You'll see: 0 → 1 → 3 → 6

5. LOCALS WINDOW

Definition:

Displays all local variables and their values in current procedure.

How to Use:

- Go to View menu → Locals Window
- Open before running code
- All variables appear automatically
- Values update live
- No setup needed

Example:

```
Sub Debug_LocalsWindow()
    Dim a As Integer, b As Integer, sum As Integer
    a = 12
    b = 7
    sum = a + b    ' See a, b, sum in Locals Window
    MsgBox "Sum: " & sum
End Sub
```

Open View → Locals Window before running.
You'll see variables update live:
a = 12
b = 7
sum = 19

6. ERROR HANDLING FOR DEBUGGING

Definition:

Catch errors without stopping – just log them.

How to Use:

- Use On Error GoTo ErrHandler
- Create error handler section
- Log error with Debug.Print
- Code continues running
- Shows error details

Example:

```
Sub Debug_ErrorHandling()
    On Error GoTo ErrHandler

    Dim x As Integer, y As Integer, z As Double
    x = 10
    y = 0
    z = x / y      ' Runtime error (divide by zero)

    MsgBox z
    Exit Sub

ErrorHandler:
    Debug.Print "Error " & Err.Number & ": " & Err.Description
    MsgBox "Check Immediate Window (Ctrl+G)."
End Sub
```

When error occurs, it's caught and logged.

Error details appear in Immediate Window.

Code does not crash.

SUMMARY TABLE - DEBUGGING TECHNIQUES

Technique	Shortcut	Purpose	Example Use
Breakpoints	F9	Pause at a line	Stop before calc
Step Into	F8	Run one line at time	Debug loops
Immediate Window	Ctrl+G	Print/log values	Debug.Print i
Watch Window	—	Track variable	Track total in loop
Locals Window	—	View all variables	Monitor a, b, sum
Error Handling	—	Catch/log errors	Print Err.Number

KEYBOARD SHORTCUTS - QUICK REFERENCE

F9	Set/Remove Breakpoint
F8	Step Into (line by line)
F5	Run full code
Ctrl+G	Open Immediate Window
Ctrl+Break	Stop running code

HOW TO DEBUG - STEP BY STEP WORKFLOW

Step 1: Open Immediate Window (Ctrl+G)
Step 2: Open Locals Window (View → Locals)
Step 3: Put cursor at Sub start
Step 4: Press F8 to step through code
Step 5: Press F9 on suspicious lines (breakpoints)
Step 6: Right-click variables → Add Watch
Step 7: Run code (F5) to see values change
Step 8: Use Debug.Print for key values

QUICK DEBUGGING CHECKLIST

- Code has Syntax Error?
 - Check red underline in editor
 - Fix spelling, quotes, keywords
 - Code has Compile Error?
 - Try to run with F5
 - Read error message
 - Fix variable declaration or type
 - Code crashes during run?
 - F9 breakpoint before error line
 - F8 step through to find problem
 - Use On Error GoTo
 - Variable has wrong value?
 - Add Watch on variable
 - Check Locals Window
 - Use Debug.Print before use
 - Loop not working?
 - Debug.Print loop variable
 - F8 step through loop
 - Check For/Next conditions
-

COMMON DEBUGGING EXAMPLES

EXAMPLE 1 - Find Wrong Calculation:

```
Sub FindWrongCalc()
    Dim a As Integer, b As Integer, result As Integer
    a = 10
    b = 5
    result = a - b ' Is this correct?
```

```
    Debug.Print "Result = " & result ' Check in Immediate Window
End Sub
```

EXAMPLE 2 - Debug a Loop:

```
Sub DebugLoop()
    Dim i As Integer
    For i = 1 To 10
        Debug.Print "i = " & i ' See all values
        If i = 5 Then
            ' Stop here with F9 to see what happens
        End If
    Next i
End Sub
```

EXAMPLE 3 - Catch Division by Zero:

```
Sub SafeDivide()
    On Error GoTo ErrorHandler
    Dim x As Double, y As Double, result As Double
    x = 100
    y = 0
    result = x / y
    Exit Sub
ErrorHandler:
    Debug.Print "Error: Cannot divide by zero"
End Sub
```

PRINT THIS PAGE FOR QUICK REFERENCE

DEBUGGING HOTKEYS:

F9 = Breakpoint F8 = Step Ctrl+G = Immediate F5 = Run

MUST-HAVE WINDOWS:

- Immediate Window (Ctrl+G)
- Locals Window (View → Locals)
- Watch Window (Right-click → Add Watch)

BASIC DEBUG CODE:

```
Debug.Print "Variable = " & variable
On Error GoTo ErrHandler
```

=====

FILE NAME: VBA_Debugging_Complete_Notes.txt

READY TO PRINT → SAVE AS PDF

=====