STUDY THIS

Debugging.

A problem in a computer program is called a **bug**. The process of getting rid of bugs is called **debugging**….so what do you do if your program does not work?

There are three different types of error that may occur…

[1] **Syntax Error**
This is when the programmer (you!) breaks the rules of the syntax of the language. For example, you may spell an instruction or property incorrectly…

Example : txtMessage.Txet instead of txtMessage.Text

Syntax errors are usually picked up by the compiler before the program is run.

[2] **Logical Error**
The program runs fine…but gives the wrong results.

Example : The program may add a discount amount instead of subtracting it.

[3] **Run-time Error**
The program compiles fine, but an error occurs when the program is run.

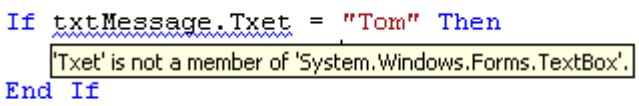
Example : The program may try to open a file of data that is not in the expected place.

Another example : trying to divide a number by 0 may cause a run-time error.

Finding Errors

To fix your errors, you first have to find them.

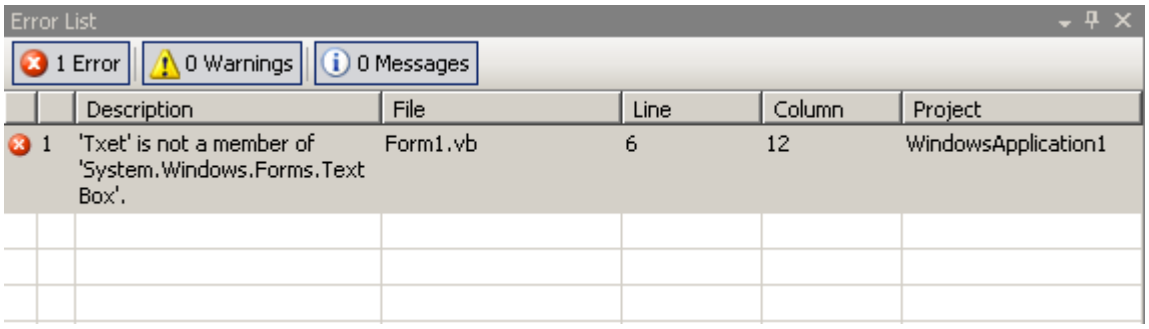
Syntax errors in VB are usually shown by a blue squiggly line. Hover your mouse cursor over the error and a helpful diagnostic error message should be displayed….



For beginners, some of these error messages take some getting used to! – but they should at least give you a clue about what the error is.

Logical errors can be much harder to track down…

There will also be an entry in the 'Error List' box if it is displayed at the bottom of the screen.

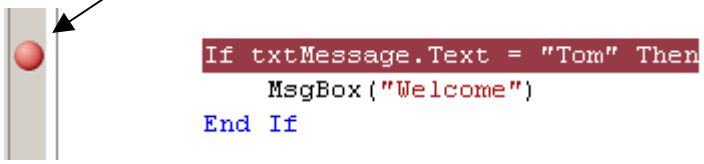


If you double-click on the error, it will take you to where it is. This is useful in a long program.

Breakpoints

If a program is not working it is possible to stop the program running at a specified line of code. To do this you need to insert a **Breakpoint**.

Click on the grey border on the left edge of the line where you want the program to stop running. A Red marker will appear and the line of code will be highlighted in red. (To remove the breakpoint – click it again)



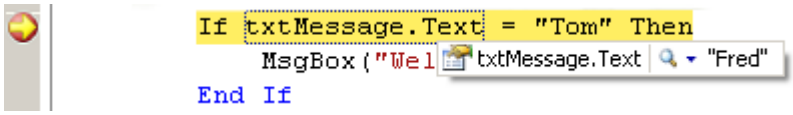
Run the program and execution will stop at this line. The line is highlighted in yellow and an arrow placed in the margin to show the line at which the program stopped.

You can now do one of two things…

[1] Check the values of variables or object properties.
[2] Single-step through the program, running one line at a time.

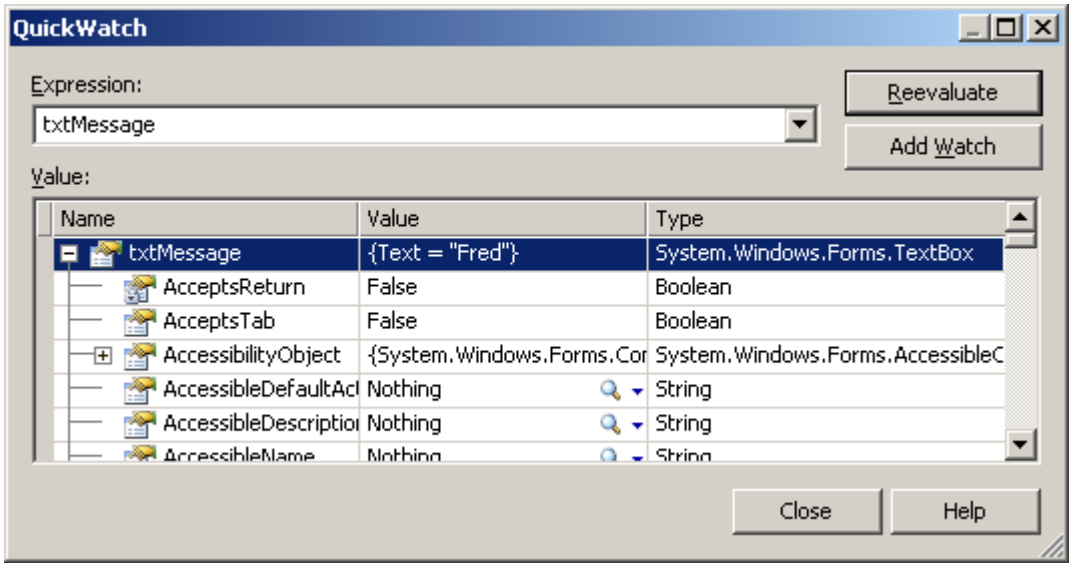
Checking Values of Variables or Object properties

When the program execution stops at a breakpoint, you can place the mouse cursor over a variable, and the value will be displayed…



Is it the value you expected it to be? If not, it may give you a clue as to what the problem is…

You could also add a Watch. (Use the Debug window). This would list the values of all the properties of an object…
Click on the 'Add Watch' button to display this in the Watches window at the bottom of the screen.



You can also add expressions (such as txtMessage.Text = "Tom") to the Watch window…to see if they are TRUE or FALSE.

You can add as many watches as you need.

Single-stepping

Use the Step Into button  to execute the next program statement. (The line highlighted in yellow is the **NEXT** line to be executed.)

Keep an eye on the values of your watches as each line is executed and it should give you a clue about what the problem is.

HINTS : If your program is not working…

1. Place a breakpoint at the start of the section.
2. Add Watches to look at values of object properties.
3. Single-step through the program – keeping an eye on your watches.