2.2.2

Write Readable Code

Readable code is code that is clear and well-documented. Writing readable code helps not just the person reading it, but also the programmer who is writing it in the first place—especially if he or she needs to take a break from the project for any period of time. You may already know from firsthand experience that stepping away from a project, even for a short time, can cause you to forget everything you worked on! Readable code helps us get up to speed more quickly. This is something to keep in mind as we work on our project for Steve. He might ask us to change or revisit the code in the future, so let's focus on writing clean, readable code. How do we do that? Let's find out.

VBA doesn't care much what your code looks like, as long as everything is typed correctly and in the right order. In general, however, code is read more often than it is written, so it helps to document and format your code to improve its overall readability. Documenting code is done by adding comments. Formatting code involves the use of whitespace.

Comments

Take a look at the code we just wrote:

Sub DQAnalysis()

Worksheets("DQ Analysis").Activate

Range("A1").Value = "DAQO (Ticker: DQ)"

Cells(3, 1).Value = "Year"

Cells(3, 2).Value = "Total Daily Volume"

Cells(3, 3).Value = "Return"

End Sub

This code works great, but it isn't self-explanatory. Also, as the code becomes more complicated (which it will), it will be useful to insert comments to explain what the code is doing. Comments can also be used to mark sections of code in order to make them easier to find.

IMPORTANT

Every programming language has some way of adding comments. If there's a tricky line of code that is difficult to understand, adding comments can help the next person reading your code understand what your code is doing—and that person might be you. Adding comments to our code helps us remember our thought process, especially after taking a break from a project for any length of time.

In VBA, comments start with a single quote. Anything you write after the single quote gets ignored, which allows us to add notes to our code. Let's explain with a comment what the three Cells lines are doing.

Above the first Cells line, insert a single quote followed by the text "Create a header row."

Sub DQAnalysis()

Worksheets("DQ Analysis").Activate

Range("A1").Value = "DAQO (Ticker: DQ)"

'Create a header row

Cells(3, 1).Value = "Year"

Cells(3, 2).Value = "Total Daily Volume"

Cells(3, 3).Value = "Return"

End Sub

You just wrote your first comment—nice work!

Whitespace

Readable code is also well organized. Enter whitespace, which refers to the use of spaces, tabs, and line breaks. Whitespace is used to organize code.

Whitespace is "invisible" to VBA. When VBA runs the subroutine you've written, it translates all your keywords and statements into instructions your computer can digest and completely ignores whitespace—with the exception of spaces between keywords and line breaks. Therefore, you can add as many spaces, tabs, and line breaks as you want, and your macro will still run the same way.

NOTE

Discussions about how to format code can get very heated—seriously! Programmers still argue about whether tabs or spaces should be used to indent code. The correct answer is, quite simply, to be consistent. Code is easiest to read and understand when it follows consistent formatting.

We'll be using whitespace to organize our code. For example, in our macro, all the code inside the subroutine is indented. This way, it's easier to see that the code belongs to this particular subroutine. Later, when we get into more complicated program flow, we'll have code blocks within code blocks, so we'll indent multiple times to keep the code organized. As programs get longer and more complicated, well-formatted code makes life so much easier.

Go ahead and test your code as often as you'd like. It's often helpful to run code as you build it and check for errors as you go. If you need to reset the worksheet after running your macro, resetting it is pretty easy, too.

First, make sure you save your macro in a text file (a text editor such as Notepad or even Google Docs or Microsoft Word will work, as long as you keep a copy of your code).

Then, close your Excel workbook without saving, then open it again. The data is once again fresh and ready for analysis. The code you saved in a text editor can now be copied back into the VBA code editor if needed.