2.5.3 Measure Code Performance

In the future, Steve may want to perform his analysis on larger datasets, and he wants to know how fast his VBA code will compile the results. To help Steve, we need to add a script that will calculate how long the code takes to execute and output the elapsed time in a message box.

To get the amount of time it will take to run Steve's `AllStocksAnalysis` script, we need to capture the start time and end time of the executed code. Lucky for us, VBA has a `Timer` function!

Get the Start and End Time

To get the start and end time we will need to initialize two variables, `startTime` and `endTime`, and then set each variable equal to the `Timer` function.

In the AllStocksAnalysis script, add the `startTime` and `endTime` variables as `Single` data types underneath the `Sub AllStocksAnalysis()` subroutine.

Next, underneath the `yearValue` variable set the `startTime` variable equal to the `Timer` function, which will allow us to start the clock!

```
Sub AllStocksAnalysis()
    Dim startTime As Single
    Dim endTime  As Single

    yearValue = InputBox("What year would you like to run the analysis on?")

       startTime = Timer
```

In the code above, we are setting the `startTime` variable equal to the `Timer` function after the `yearValue` variable because we want to start the clock after we have entered the year in the `InputBox()` command.

Next, scroll to the end of your `AllStocksAnalysis` script. After the last `Next i` and before the `End Sub` command, set the `endTime` variable equal to the `Timer` function. Finally, create a message box statement that will tell us how long the code took to run by subtracting the `startTime` from the `endTime` for the analysis.

```
Next i

    endTime = Timer
    MsgBox "This code ran in " & (endTime - startTime) & " seconds for the y

End Sub
```

Displaying the Elapsed Time

When you run the macro for both years, a message box similar to the images below will pop up and show the elapsed time for 2017 and 2018.

NOTE

> The first time you run a macro, the elapsed time may be longer than subsequent runs because computer resources need to be allocated to run the macro. Once allocated, these resources are ready for subsequent runs.

- [ ]
- [ ]

DEEP DIVE

Congratulations! Steve is ecstatic that he can see how long the scripts take to run.

ADD/COMMIT/PUSH

You know the drill! Push your changes to the “stocks-analysis” repo in GitHub.