

The Yahoo Key Statistics Function

Professor

Nico Fermin Cota

Date Submitted

March 16, 2009

Group Number

Six

Team Members

Ezra van Gelder	250 199 773
Joel Belisle	250 273 831
Stephanie Puzio	250 262 512
Owen Blackwell	250 249 284
Shawn Feldon	250 279 332

Introduction

Yahoo Finance is a great resource to find key statistical information on publicly traded companies including market capitalization and share information, various trading multiples, and other company characteristics. As a result of its comprehensiveness, it is a very valuable tool for finding information about a particular company. Given the output format of information, a user-defined function can be developed to organize and display the data in a useful format.

Describing Key Aspects of the UDF

Defining the Input Values

YAHOO_KEY_STATISTICS_FUNC (*Function*) requires only one input: TICKERS_RNG. The TICKERS_RNG is the list of company ticker symbols that the user wants to extract company information about from the Yahoo site. In other words, the *Function* will search for each ticker in the website and return the corresponding statistics.

Key Questions for Pre-Parsing Code

1) Explain Logic of If Statement

Initially the Function begins by defining various user variables that will be used throughout the code; from counting variables (h, i, j, and k) to various strings, vectors, and matrices. Next, the Function ensures that the input, TICKERS_RNG, is in the correct format and stores it in the function variable TICKERS_VECTOR. In order to ensure that the input is in the proper format, the following if statement is used:

```
If IsArray(TICKERS_RNG) = True Then
    TICKERS_VECTOR = TICKERS_RNG
    If UBound(TICKERS_VECTOR, 1) = 1 Then
        TICKERS_VECTOR = MATRIX_TRANSPOSE_FUNC(TICKERS_VECTOR)
    End If
Else
    ReDim TICKERS_VECTOR(1 To 1, 1 To 1)
    TICKERS_VECTOR(1, 1) = TICKERS_RNG
End If
```

First, the code checks to see if the input is an array or a single value. If it is an array then it is stored to TICKERS_VECTOR, which was defined as a variant and thus, ready to accept the array. Then the code checks to see if the array is a column or a row vector. If it is a row vector (i.e. the number of rows found from UBound(TICKERS_VECTOR, 1) is 1), then the MATRIX_TRANPOSE_FUNC is called and passed the TICKERS_VECTOR in order to change it to a column vector (i.e. one column with n rows). Finally, it is restored in the TICKERS_VECTOR variable as the new column vector. However, if

the input was not an array, but a single company ticker, then the TICKERS_VECTOR is re-dimensioned to be a single value (1 row and 1 column) and set equal to the given ticker symbol.

2) Why is NROWS = 58?

The NROWS column was initialized to the static value of 58 because we know Yahoo always returns 58 statistics for each stock (58 headers). The site has returned 58 stats since January 31st, 2008 and will continue to do so unless Yahoo initiates a site resign. If the site is changed then the function can be altered by changing this one value.

Although it seems counter-intuitive, the code did not confuse NROWS and NCOLUMNS. The reason for having 58 rows and a column for each ticker is due to the way the data will be retrieved and stored for each company. First, each company's key statistics are retrieved and stored in a column vector (TEMP_GROUP), which is transferred to DATA_GROUP that has a similar structure. The data is finally transposed in one of the last For-Next Loops (see Q27); therefore, the final matrix output has a row for each ticker and 59 columns (a symbol column and 58 statistics columns).

3) Why use delimiter character = "," ?

Although the original source code downloaded from Yahoo is not split using commas, the DELIM_STR is set equal to a comma because it is added to the retrieved information to act as a delimiter character for parsing later in the code. The DELIM_STR is added to the code to organize each of the statistics for each company. The retrieved data for a given ticker is stored in TEMP_GROUP, with each row holding one key stat: the header title and the corresponding data statistic separated by a comma. The array is transferred to the DATA_GROUP array which will ultimately store all the ticker stats. Finally, the headers and stats data is parsed via the DELIM_STR and stored in TEMP_MATRIX. Secondly, the DELIM_STR is also used to eliminate all the commas in the original source data (see Q9).

4) What is the purpose of using the array DATA_GROUP?

The DATA_GROUP array is the variable that will be used to store the retrieved and concatenated data for all the companies. It is resized to the number of individual ticker values inputted into the function (NCOLUMNS). Once all the data for a single ticker is retrieved and stored in the TEMP_GROUP array, the whole array is transferred into one of the locations of the positions in DATA_GROUP and then the *Function* moves onto the next ticker symbol.

5) Why do I need TICKER_STR?

TICKER_STR is used to store and identify the current stock with which the function is working. It is also used to download the data (see Q7). Coding the ticker symbol is a more reliable method to track which company is being processed compared to relying on the position in TICKER_VECTOR by using TICKER_VECTOR(h,1).

6) What is the purpose of setting REFER_STR to “yfnc_datamodoutline1” string? (See Q10)

The string “yfnc_datamodoutline1” is used to identify the beginning location of the key stats information from the downloaded source data.

The string “yfnc_datamodoutline1” is also class name. Classes are a form of cascading style sheet (CSS), used to style data. The “yfnc_datamodoutline1” is **unique** to the data tables and is therefore the best method of identifying their beginning. For example, the table titles such as “VALUATION MEASURES” or “FINANCIAL HIGHLIGHTS” may be present elsewhere on the page, in a menu item or dynamic advertisement. Since “yfnc_datamodoutline1” is unique to the data tables it provides an efficient way to identify the data which YAHOO_KEY_STATISTICS_FUNC requires.

7) Why do we need SRC_URL_STR?

The source URL is required in order to retrieve the source code of from the Yahoo web pages. SRC_URL_STR has both static and dynamic components.

```
SRC_URL_STR = "http://finance.yahoo.com/q/ks?s=" & TICKER_STR
```

The static component is “http://finance.yahoo.com/q/ks?s=” remains unchanged, while the TICKER_STR is appended to retrieve the current stock’s web page and is changed each time the loop iterates.

8) What is the purpose of retrieving the data and saving it in the PUB_WEB_DATA_PAGES_MATRIX array?

The PUB_WEB_DATA_PAGES_MATRIX function retrieves and stores the data in a synchronous and easily accessible manner by converting it to an array.

9) Why am I replacing all of these characters with blanks?

After the original source code is downloaded from Yahoo the code loops through and replaces all linefeeds, colons, commas, and “&s” with blanks. Table 1 summarizes the reasons for these replacements.

Table 1: Replacements in Source Code

CODE	EXPLANATION
<code>Replace(DATA_STR, Chr(10), "")</code>	The initial data code downloaded from the Yahoo site is approximately 25 pages long and replacing the linefeeds with blanks reduces much of the code space
<code>Replace(DATA_STR, ":", "")</code>	All colons are replaced with blanks to eliminate any potential errors that may arise when searching and retrieving the data and/or in one of the final If statements when a colon is added if only one ticker is present: <code>TEMP_MATRIX(i, 1) = TICKER_STR & ":" & Trim(Mid(TEMP_STR, 1, j - 1))</code>
<code>Replace(DATA_STR, DELIM_STR, "")</code>	This replacement is used to eliminate any commas in the source code as they will be inserted in a later loop and used to help parse the retrieved data.
<code>Replace(DATA_STR, "&"; "&")</code>	"&" is html code used to represent the "&" symbol. It is replaced so that the ampersand will display correctly as "&" instead of "&".

10) Why do I use the following routine: If i = 0 Then Goto 1983?

If $i = 0$, this suggests VBA wasn't able to successfully find REFER_STR (see Question 6), implying the web page had one of the following issues:

- did not load successfully;
- has been reformatted using a different CSS;
- has been redesigned; or
- does not have the data for the given ticker.

By skipping the code to 1983, the loop is essentially told to restart with the next ticker.

11) What is the purpose of the new REFER_STR string? (See Q12 below.)

REFER_STR is replaced with the following string:

```
REFER_STR = _
"</spacer></td></tr></table></td></tr></table>"
```

The new string marks the 'the beginning of the end' of the company statistics and information in the source data. Similar to the initial REFER_STR value, it is unique: it only exists once in the source data.

12) Explain this routine: Mid(DATA_STR, i, j - i + Len(REFER_STR))?

At this point in the loop, i stores the starting position of the first REFER_STR (beginning of stats data in the source code) and j stores the starting position of the second REFER_STR (the end of the stats data in the source code). Therefore, the Mid function extracts the data in DATA_STR between these two points: the company data.

Essentially, the Mid function eliminates the unnecessary header and footer code resulting in only 8 pages of source code. This avoids unexpected changes in the header and footer code from affecting data parsing as well as increases speed and efficiency of the code execution.

13) Can you explain the purpose of REFER_STR = "</td></tr></table></td></tr></table>
</td></tr><tr><td"?

The new REFER_STR marks the exact location after the last data statistic value (for Last Split Date) Therefore, REFER_STR is used as the check for the data extracting loop. The *Function* will loop until the string between the location of j plus the length of REFER_STRING (which is 57) is equal to REFER_STR (i.e. until REFER_STR is reached.)

```
Loop Until Mid(DATA_STR, j, Len(REFER_STR)) = REFER_STR
```

The information after this relates to other Yahoo functions (e.g. Add this stock to my Portfolio), formatting, and references.

14) Can you elaborate why I use TEMP_GROUP as an array?

TEMP_GROUP is an array because it is used to store the retrieved and reformatted data from the main loop (see Q15-Q25). After each statistic for a company is retrieved it is stored in its concatenated form using the code below:

```
'ReDim Preserve TEMP_GROUP(1 To k)
'TEMP_GROUP(k) = TEMP_STR
'(and after the loop:) DATA_GROUP(h) = TEMP_GROUP
```

The ReDim Preserve code ensures that any data previously stored in the array (from previous loops) is saved. The array is extended by one data point for each iteration, with the new key statistic header and data is stored in the added location.

Key Questions for Parsing Code

The next portion of code, the DO Loop, is essentially the parsing code for the various statistics and is explained in questions 15 to 25.

```
j = 1
k = 0
Do
  i = InStr(j, DATA_STR, "yfnc_tablehead1") + Len("yfnc_tablehead1") + 1
  i = InStr(i, DATA_STR, ">") + 1
  j = InStr(i, DATA_STR, "<")

  If j = 0 Then: Exit Do

  TEMP_STR = Mid(DATA_STR, i, j - i) & DELIM_STR
  -----
  i = InStr(j, DATA_STR, "yfnc_tabledatal") + Len("yfnc_tabledatal") + 1

  If Mid(DATA_STR, i, 6) = "><span" Then: i = i + 6

  i = InStr(i, DATA_STR, ">") + 1
  j = InStr(i, DATA_STR, "<")
  If j = 0 Then: Exit Do
  TEMP_STR = TEMP_STR & Mid(DATA_STR, i, j - i)
  -----

  k = k + 1
  ReDim Preserve TEMP_GROUP(1 To k)
  TEMP_GROUP(k) = TEMP_STR

  If k > NROWS Then: Exit Do

Loop Until Mid(DATA_STR, j, Len(REFER_STR)) = REFER_STR!
```

15) Why does j = 1?

The variable j is set to the starting position at which the *Function* starts looking in the DATA_STR, which now stores only the source code for the key statistics of TICKER_STR.

16) Why does k = 0?

The variable k is initialized to 0 (when beginning the loop for each ticker symbol) because at the end of the loop k is incremented by 1 and used to store the retrieved data into the corresponding position in TEMP_GROUP. Most importantly, k is used as an exit threshold for the DO Loop by checking if k is larger than NROWS (the number of stats being retrieved). Alternatively, k could be initialized to 0 and the loop could begin by incrementing k, but this would cause an error at the exit threshold as it would loop one additional time (a 59th time) because when k = 58, it is still not larger than NROWS and would loop again.

17) What am I trying to accomplish with i?

In this case, i is used to locate the positions of the various table headings in the DATA_STR. The function first locates the position of the text *yfnc_tablehead1* and then searches for the next ">" sign and adds one. This represents the starting position of the first header (or data point). For example, the data code for the first key statistic is: "yfnc_tablehead1" width="75%">Market Cap (intraday)<.

Therefore, the first loop would return i = 30, which is the position just before the M.

18) Why am I looking for "<"?

The position of the "<" represents the end of each header or data point. The data is given by ">data<". This allows the function to loop through the DATA_STR, find these characters, and extract the found between them. Therefore, in the example above, j would equal 51.

19) Why do I use the following routine: If j = 0 Then Exit Do?

If j = 0, that means that the InStr function could not find the "<" character. InStr returns 0 if the character cannot be found. If this is the case, Yahoo either changed their html source code, or there is an error in the code that downloads the data and stores it in the DATA_STR array.

20) Can you explain this line: Mid(DATA_STR, i, j - i) & DELIM_STR?

The Mid function is used to extract the stat header from the DATA_STR array, starting in the first position after the ">" sign and ending at the first position before the "<" sign. The code then adds the DELIM_STR (",") to the end of the data point, in order to be able to add the data for the stat after it.

21) What I am looking for with i = InStr(j, DATA_STR, "yfnc_tabledata1") + Len("yfnc_tabledata1") + 1?

The InStr function now starts at the j position (ending position of the first heading) within the DATA_STR and looks for the data string indicator, *yfnc_tabledata1*. Once found, the length of the string was added to the position and the new position is stored as i. The i position will be used to locate the starting position of the corresponding data point.

22) Why does i = i + 6 if Mid(DATA_STR, i, 6) = "><span"?

This line of code allows the function to check if the string "><span" exists after the *yfnc_tabledata1* in the html source code. The check is necessary because it was already determined that that data lies between ">...<", therefore, the section of text would improperly locate the starting position of the data just before the "><span" string. Thus, the data point would be returned as ">". The check enables the loop to bypass the extra "><" characters and properly locate the data statistic.

23) What is the purpose of Mid(DATA_STR, i, j - i)?

This line of code extracts the data statistic (whatever was located between the ">...<" characters) to the TEMP_STR array. After the first run through the loop, this line of code would return "137.63B".

24) Why am I concatenating TEMP_STR with Mid(DATA_STR, i, j - i)?

The code concatenates TEMP_STR (the statistic header and comma) with the actual data stat in order to store the two as a single string (header and data) in the TEMP_GROUP(k) position. Therefore, the first position for JNJ would be "Market Cap (intraday),137.63B".

25) Why do I exit the Do Loop if k > NROWS?

If k > NROWS this means that all 58 statistics have been retrieved and stored in the TEMP_GROUP array. Since Yahoo always stores 58 (NROWS) statistics for each stock and since each entry in the TEMP_GROUP represents one company, the loop must stop when it reaches data point 58. The function will then increase the h counter which represents the number of stocks being analyzed (NCOLUMN), and repeat the parsing/storing process.

26) Can you explain what am I trying to accomplish here? (See code)

```
If NCOLUMNS = 1 Then
  ReDim TEMP_MATRIX(1 To UBound(TEMP_GROUP), 1 To 2)
  For i = 1 To UBound(TEMP_GROUP)
    TEMP_STR = TEMP_GROUP(i)
    If TEMP_STR = "" Then: GoTo 1984
    j = InStr(1, TEMP_STR, DELIM_STR)
    TEMP_MATRIX(i, 1) = TICKER_STR & ": " & Trim(Mid(TEMP_STR, 1, j - 1))
    TEMP_MATRIX(i, 2) = CONVERT_STRING_NUMBER_FUNC(Trim(Mid(TEMP_STR, j +
1, Len(TEMP_STR) - j)))
```

The first section of the If statement is used for the case when the number of columns (number of ticker symbols) is equal to 1. In this case, the TEMP_MATRIX is resized to have 58 rows (from NROWS) and 2 columns (heading, data). Next, a For-Next loop is used to run through the TEMP_GROUP array, splitting up the header from the data point for each "(i)" entry. This is accomplished by splitting each "(i)" entry in the TEMP_GROUP by the DELIM_STR (,) which was inserted by a previous loop. The InStr function is used to locate the position of each comma, and the Mid function is used to take the text before and after the comma and insert the data into the final TEMP_MATRIX in positions (i, 1) and (i,2).

Handling the case with only one ticker symbol separately is unnecessary and there are a few important lines of code that are missing from this segment to print the output matrix properly. First, the TEMP_MATRIX should be resized to include a position for the "Symbol" heading. The YAHOO_KEY_STATISTICS_FUNC writes the data into an array with the statistics in rows and the ticker symbols in columns. This matrix is then transposed, reformatted, and printed in the destination worksheet. If there is only one stock in the TICKERS_RNG, the output data is not

transposed and not reformatted properly. The following lines of code fill the TEMP_MATRIX with the data from the TEMP_STR, with each row representing a new statistic. This results in a 58 X 2 matrix printed into the excel worksheet instead of the 2 X 59 (58 + 1 for the “Symbol” heading) output matrix for cases with more than one input stock.

Finally, the code doesn’t split the ticker symbol from the data point, so each statistic includes the ticker symbol: data point (ex. JNJ: Market Cap (intraday)). If this section of code is eliminated, the program will run without the above inconsistencies.

The impact of this coding is shown in **Exhibit 1**.

27) Elaborate in this section of the If statement. (See code)

```
Else
  ReDim TEMP_MATRIX(1 To NCOLUMNS + 1, 1 To NROWS + 1)
  TEMP_MATRIX(1, 1) = "Symbol"

  For h = 1 To NCOLUMNS
    TEMP_MATRIX(1 + h, 1) = TICKERS_VECTOR(h, 1)
    If IsArray(DATA_GROUP(h)) = False Then: GoTo 1986
    TEMP_GROUP = DATA_GROUP(h)
    For i = 1 To NROWS
      If i > UBound(TEMP_GROUP) Then: Exit For

      TEMP_STR = TEMP_GROUP(i)
      If TEMP_STR = "" Then: GoTo 1985

      j = InStr(1, TEMP_STR, DELIM_STR)
      If j = 0 Then: GoTo 1985
      k = InStr(1, TEMP_STR, "(")
      If k = 0 Then: k = j

      TEMP_MATRIX(1, i + 1) = Trim(Mid(TEMP_STR, 1, k - 1))
      REFER_STR = Trim(Mid(TEMP_STR, j + 1, Len(TEMP_STR) - j))

      If REFER_STR = "N/A" Or REFER_STR = "NaN%" Or REFER_STR = "NaN"
Then
      TEMP_MATRIX(h + 1, i + 1) = 0
      Else
      TEMP_MATRIX(h + 1, i + 1) =
CONVERT_STRING_NUMBER_FUNC(REFER_STR)
      End If
```

The Else portion of the above If statement handles the case where more than one stock is being analyzed (TICKERS_RNG is an array). First, the TEMP_MATRIX is resized to add an extra position for the ticker heading (“Symbol”). The first position (1,1) in the TEMP_MATRIX is then set equal to “Symbol”. The first For-Next loop will step through the number of stocks being analyzed. Since the “Symbol” heading has been added to the TEMP_MATRIX, the next line of code shifts all of the data

in the TICKERS_VECTOR by one position. If the DATA_GROUP(h) containing the header and data points for each stock is not an array (no data), the loop will move on to the next stock.

The loop then steps through the 58 data points for each stock, separating the heading from the data point and placing them in two different positions in the TEMP_MATRIX. This is accomplished by using the InStr function, searching for the DELIM_STR (“,”) character, and cutting the data into two components based on this position. The TEMP_MATRIX is then filled with headings and the corresponding data in columns in the worksheet; with each row representing a different stock. The data points are then run through the CONVERT_STRING_NUMBER_FUNCTION which converts the data into its proper format (dates, numbers).

Conclusion

The *Function* downloads many pages of data from Yahoo Finance and parses into a logically formatted, easily analyzed format. This creates the potential to almost instantaneously download enormous amounts of financial data so that the majority of time can be spent on analyzing the data as opposed to acquiring it.

Exhibit 1

Effect of IF Statement on One Ticker Input

IF Statement Included

[blank]	JNJ:Market Cap (intraday)	JNJ: Enterprise Value (15-Mar-09)	JNJ: Trailing P/E (ttm intraday)	JNJ: Forward P/E (fye 28-Dec-10)	JNJ: PEG Ratio (5 yr expected)
[blank]	140060000	139100000	11.09	10.36	1.36

The IF statement modifies the headings and removes the first column "Symbol" that appears when there are multiple tickers. As well, the information is presented in two columns as opposed to two rows as shown above.

IF Statement Excluded

Symbol	Market Cap	Enterprise Value	Trailing P/E	Forward P/E	PEG Ratio
JNJ	140060000	139100000	11.09	10.36	1.36

Removing the IF statement (for example, by commenting it out of the code) results in the above output. It is entirely consistent with the output for multiple tickers resulting from the IF statement and is given in two rows as shown above.

Note: Only the first six headings were included.