Contents

Stock Market Functions Add-In Overview 2

Documentation for RCHGetElementNumber() function 5

Documentation for smfGetAdvFNElement() function 8

Documentation for RCHGetYahooQuotes() function 10

Documentation for smfGetOptionQuotes() function 14

Documentation for smfGetYahooOptionQuote() function 20

Documentation for smfGetMSNOptionQuote() function 22

Documentation for smfGetOXOptionQuote() function 24

Documentation for smfGetOptionExpiry() function 26

Documentation for RCHGetYahooHistory() function 28

Documentation for smfPricesBetween() function 31

Documentation for smfPricesByDates() function 34

Documentation for RCHGetHTMLTable() function 36

Documentation for RCHGetTableCell() function 38

Documentation for RCHGetWebData() function 42

Documentation for smfGetCSVFile() function 44

Documentation for smfGetEconData() function 46

Documentation for smfGetTagContent() function 48

Documentation for RCHCreateComment() function 50

Documentation for smfTech() function 52

Documentation for smfStrExtr() function 54

Stock Market Functions Add-In Change Log 55

Stock Market Functions Add-In Overview

The Stock Market Functions (SMF) add-in is a collection of user-defined functions (UDF) that can allow you to retrieve data from web pages and web queries via EXCEL formulas. To use an add-in, you need to download it to your computer and install it (a simple process). For general information on add-ins, refer to:

http://www.bettersolutions.com/excel/ECA723/LT423111411.htm

How do I install the add-in?

First step would be to create this directory on your computer:

C:\Program Files\SMF Add-In

Next, you'd download the current ZIP archive of the add-in from this group's files area into that directory and extract all of the files from the ZIP archive there. Then, you'd need to tell EXCEL to use it. In EXCEL 2002, that process is:

1. Open a new workbook
2. Use menu option > Tools > Add-Ins > Browse
3. Browse to the XLA file that was extracted from the ZIP archive and double-click on it

That should be all you need to do.

Note -- the reason a non-standard add-in path was used above is to ease sharing of workbooks that use the add-in. Because of how EXCEL saves UDF (User Defined Function) formula usage, sharing works best if the add-in is located in the same location on both the computer that created the workbook and the computer that is opening the workbook. If you don't use the recommended directory above, the add-in will still work, but you will have to manually update workbook links when you open any workbook created by someone who had their version of the add-in located elsewhere. Also, older versions of EXCEL treat add-in locations differently, so they may require you to manually locate the add-in when opening up a workbook created on another machine. In many cases, saving the workbook on your machine first may ease the resolution of this location problem.

Here are several illustrated descriptions of how to install an EXCEL add-in:

http://www.bettersolutions.com/excel/ECA723/LD023821888.htm
http://www.dslimited.biz/excel_tutorials/installadd-in.html

What does the add-in do?

There are five primary user-defined functions that the SMF add-in provides:

- =RCHGetElementNumber() is a UDF that returns a specific data element from a specified data source (i.e. web page). For example, this formula would return the Market Capitalization amount from Yahoo's Key Statistics page for ticker "IBM":

  =RCHGetElementNumber("IBM", 941)

  More detailed documentation on the RCHGetElementNumber() function can be found here: 2.0 - RCHGetElementNumber.html

- =RCHGetTableCell() is a UDF that extracts a specific data element from a specified web page. Most RCHGetElementNumber() functions are actually just an RCHGetTableCell() function with the parameters saved and assigned to an element number. For example, this formula would return the Market Capitalization amount from Yahoo's Key Statistics page for ticker "IBM":

  =RCHGetTableCell("http://finance.yahoo.com/q/ks?s=IBM",1,">Market Cap")

  More detailed documentation on the RCHGetTableCell() function can be found here: 5.1 - RCHGetTableCell.html

- =RCHGetYahooQuotes() is a UDF that returns delayed stock quotes and other data from the Yahoo quotes interface. For example, this formula would return a range of data showing the last trading price, the price/sales ratio, and the price/book ratio for "IBM" and "MMM":

  =RCHGetYahooQuotes("IBM,MMM", "l1p5p6")

  More detailed documentation on the RCHGetYahooQuotes() function can be found here: 3.0 - RCHGetYahooQuotes.html

- =RCHGetYahooHistory() is a UDF that returns historical stock quotes from the Yahoo historical quotes interface. For example, this formula would return a range of data containing historical quotes data from 6/1/2006 thru 6/16/2006 for "IBM":

  =RCHGetYahooHistory("IBM", 2006, 6, 1, 2006, 6, 16)

  More detailed documentation on the RCHGetYahooHistory() function can be found here: 4.0 - RCHGetYahooHistory.html

- =RCHGetHTMLTable() is a UDF that is used to extract an HTML table from a web page. For example, this formula would return a range of data containing a table from

Yahoo's Key Statistics page:

=RCHGetHTMLTable("http://finance.yahoo.com/q/ks?s=MMM","Market Cap (intraday)",-1,"",1)

More detailed documentation on the RCHGetHTMLTable() function can be found here: 5.0 - RCHGetHTMLTable.html

If you check the "Documentation" folder of the files area of the group, you'll find documentation on a number of other functions. Also, the "Links" area of the group contains links to messages with frequently asked questions and tips on using the add-in.

Documentation for RCHGetElementNumber() function

Description

Returns a specific data element from a specified data source (i.e. web page).

Syntax

=RCHGetElementNumber(ticker , elementnumber [, errorvalue])

Parameters

ticker = A ticker symbol indicating which company data is to be returned for. In addition, there are several literals that can be specified for this parameter to request other information. See the "Examples" section for more details.

elementnumber = A number specifying which data element is to be retrieved for a ticker symbol. A list of element numbers and the data sources and data elements they are related to can be found in the RCHGetElementNumber-Element-Definitions.xls workbook distributed with the add-in.

errorvalue = A string or numeric value to be returned if there is an error in finding the data element. A default value of "error" is used if nothing is passed. This can prevent needing to put IF() statements in a cell to make a display or calculation of items easier to read.

Examples

Example Formula	Value Returned	Description
=RCHGetElementNumber("Version")	Stock Market Functions add-in, Version 2.0j	Tells you which version of the add-in you are using.
=RCHGetElementNumber("Source", 941)	YahooKS	Gives you the data source for element number 941
=RCHGetElementNumber("Element",941)	Market Cap	Gives you the element name for element number 941
=RCHGetElementNumber("Web Page",941)	http://finance.yahoo.com/q/ks?s=~~~~~	Gives you the web page

		element number 941 is retrieved from ("~~~~~" is the ticker symbol)
=RCHGetElementNumber("IBM", 941)	119,670,000	Market Capitalization from data source "YahooKS" for ticker "IBM"
=RCHGetElementNumber("IBM", 1541)	119,670,000,000	Market Capitalization from data source "MSN" for ticker "IBM"
=RCHGetElementNumber("P-URL",941)	http://finance.yahoo.com/q/ks?s=~~~~~	RCHGetTableCell() function parameter
=RCHGetElementNumber("P-Cells",941)	1	RCHGetTableCell() function parameter
=RCHGetElementNumber("P-Find1",941)	VALUATION MEASURES	RCHGetTableCell() function parameter
=RCHGetElementNumber("P-Find2",941)	MARKET CAP	RCHGetTableCell() function parameter
=RCHGetElementNumber("P-Find3",941)		RCHGetTableCell() function parameter
=RCHGetElementNumber("P-Find4",941)		RCHGetTableCell() function parameter
=RCHGetElementNumber("P-Rows",941)	0	RCHGetTableCell() function parameter
=RCHGetElementNumber("P-End",941)	0	RCHGetTableCell() function parameter
=RCHGetElementNumber("P-Look",941)	0	RCHGetTableCell() function parameter

=RCHGetElementNumber("P-Type",941)	0	RCHGetTableCell() function parameter

Usage notes

- When elements are retrieved, the source code of the web page is saved in an array within the VBA environment, then the element is extracted from that source code. It is done that way so that when a number of elements are retrieved from the same web page, the web page only needs to be retrieved once. Note that the array to save web pages is limited to 1000 pages. Anything beyond that will get an error message. The entire array can be reset by either exiting and re-entering EXCEL or by running the smfForceRecalculation macro. This macro purges all saved web pages and reacalculates all functions.

- ...coming later...

Documentation for smfGetAdvFNElement() function

Description

Returns a specific data element from the annual and quarterly financial statements web pages on AdvFN.

Syntax

=smfGetAdvFNElement(Ticker, Period, #Cells, Find1, [Find2], [ErrorMsg], [Type])

Parameters

Ticker = A Yahoo ticker symbol indicating which company data is to be returned for.

Period = A string value to indicate whether annual or quarterly data elements are to be retrieved.

A = Annual
Q = Quarterly

#Cells = A table cell counter for which fiscal period is to be retrieved (after function is positioned on the page by "Find1" and "Find2"). A value of 1 would retrieve the most current fiscal period, 2 would retrieve the next fiscal period, and so on.

Find1 = A string value to search for to position the function on the page before determining which fiscal period to retrieve. Using a special value of 999 will tell the function to return the number of annual or quarterly periods that AdvFN has for the ticker symbol.

Find1 = An optional string value to search for to further position the function on the page before determining which fiscal period to retrieve.

ErrorMsg = An optional string or numeric value to be returned if there is an error in finding the data element. A default value of "Error" is used if nothing is passed. This can prevent needing to put IF() statements in a cell to make a display or calculation of items easier to read.

Type = An optional integer value to determine the type of Internet request to make. Defaults to 0. For now, other values are experimental:

0 = XMLHTTP "Get" Request
1 = IE Object Request
2 = HTMLDocument Request

3 = XMLHTTP "Post" Request

Examples

- To get the four most recent year-end dates from the annual financials:

  =smfGetAdvFNElement("MMM","A",1,">Year End Date<")
  =smfGetAdvFNElement("MMM","A",2,">Year End Date<")
  =smfGetAdvFNElement("MMM","A",3,">Year End Date<")
  =smfGetAdvFNElement("MMM","A",4,">Year End Date<")

- To find out the number of annual periods that AdvFn has:

  =smfGetAdvFNElement("MMM","A",999)

Usage notes

- When elements are retrieved, the source code of the web page is saved in an array within the VBA environment, then the element is extracted from that source code. It is done that way so that when a number of elements are retrieved from the same web page, the web page only needs to be retrieved once. Note that the array to save web pages is limited to 1000 pages. Anything beyond that will get an error message. The entire array can be reset by either exiting and re-entering EXCEL or by running the smfForceRecalculation macro. This macro purges all saved web pages and reacalculates all functions.

- ...coming later...

Documentation for RCHGetYahooQuotes() function

Description

Returns delayed stock quotes and other data from the Yahoo quotes interface.

Syntax

=RCHGetYahooQuotes(tickerlist, [datacodelist], [serverid], [refresh], [headers])

Parameters

tickerlist = A list of ticker symbols to get quotes and/or other data for from the Yahoo quotes server. This parameter can either be a single ticker symbol, a comma-delimited list of ticker symbols, a cell reference, or a range reference.

datacodelist = An optional list of data codes telling the Yahoo quotes server which data elements you want. This parameter can either be a string concatenation of all desired data codes, a cell reference, or a range reference. The default value for this parameter is "sl1d1t1c1ohgv", which would return ticker symbol, last price, date and time of last price, change for the date, opening/high/low price of the day, and volume for the day. Here is a table listing the most commonly used data codes:

Basic Quote		**Fundamentals**		**Technicals**		**Estimates**		**Real-Time (ECN)**	
Code	**Description**	**Code**	**Description**	**Code**	**Description**	**Code**	**Description**	**Code**	**Description**
s	Symbol	j1	Market Capitalization	a2	Average Daily Volume	t8	1yr Target Price	k1	Last Trade (ECN with Time)
n	Name	f6	Float Shares	t7	Ticker Trend	e9	EPS Est. Next Quarter	b3	Bid (ECN)
p	Previous Close	r5	PEG Ratio	j	52-week Low	e7	EPS Est. Current Yr	b2	Ask (ECN)
o	Open	p5	Price/Sales	j5	Change From 52-week Low	e8	EPS Est. Next Year	c6	Change (ECN)

h	High	p6	Price/Book	j6	Pct Chg From 52-week Low	r6	Price/EPS Est. Current Yr	w4	Day's Value Change (ECN)
g	Low	b4	Book Value	k	52-week High	r7	Price/EPS Est. Next Yr	c8	After Hours Change (ECN)
x	Exchange	r	P/E Ratio	k4	Change From 52-week High			j3	Market Cap (ECN)
l1	Last Trade (Price Only)	e	Earnings/Share	k5	Pct Chg From 52-week High			r2	P/E (ECN)
l	Last Trade (With Time)	j4	EBITDA	m3	50-day Moving Avg			k2	Change & Percent (ECN)
d1	Date of Last Trade	s7	Short Ratio	m7	Change From 50-day Moving Avg			v7	Holdings Value (ECN)
t1	Time of Last Trade	d	Dividend/Share	m8	Pct Chg From 50-day Moving Avg			g5	Holdings Gain & Percent (ECN)
k3	Last Trade Size	y	Dividend Yield	m4	200-day Moving Avg			g6	Holdings Gain (ECN)
c1	Change	r1	Dividend Pay Date	m5	Change From 200-day Moving Avg			m2	Day's Range (ECN)
p2	Percent Change	q	Ex-Dividend Date	m6	Pct Chg From 200-day Moving Avg			i5	Order Book (ECN)

c	Change & Percent								
v	Volume	**Option Quote Additions**							
m	Day's Range	**Code**	**Description**						
j	52-week Low	o1	Open interest?						
k	52-week High	p3	Type of option						
w	52-week Range	e3	Expiration date						
b	Bid	s3	Strike price						
b6	Bid Size	n	Name of option						
a	Ask								
a5	Ask Size								

serverid = An optional parameter to specify the country id of an alternative Yahoo Quotes server. The default is a null string, which uses the normal server. From what I've seen on Yahoo, I believe these values can be specified:

Server ID	Yahoo Quotes Server For
ar	Argentina
au	Australia and New Zealand
br	Brazil
ca	Canada
cf	French Canada
chinese	Chinese
cn	China
de	Germany
es	Spain
fr	France
hk	Hong Kong
in	India
it	Italy
jp	Japan

kr	Korea
sg	Singapore
uk	UK and Ireland

refresh = An optional parameter that allows automatic refreshing of quotes via the F9 key (i.e. "recalculate"). All you need to do is pass "NOW()" as the value of this parameter to activate a refresh capability. Basically, all it does is cause a change in parameter values, which makes the function recalculate.

headers = An optional parameter that allows column headings to be generated for the returned columns of data.

Examples

- This would return the default data items for ticker symbol IBM:

  =RCHGetYahooQuotes("IBM")

- This would return updated default data items for ticker symbol IBM each time F9 was pressed:

  =RCHGetYahooQuotes("IBM",,,NOW())

- This would return default data items for ticker symbol IBM with columns headings on the returned data:

  =RCHGetYahooQuotes("IBM",,,,1)

- An step-by-step example of creating a current quotes table can be found here.

- ...more to come

Usage notes

- In most cases, this will need to be an array-entered formula. To array-enter a formula in EXCEL, first highlight the range of cells where you would like the returned data to appear -- the number of rows for the range should be AT LEAST the number of ticker symbols you are requesting from the function, while the number of columns for the range should be AT LEAST the number of data items you are requesting for each ticker symbol from the function. Next, enter your formula and then press Ctrl-Shift-Enter.

- ...more to come

Documentation for smfGetOptionQuotes() function

Description

Returns delayed quotes and related data for options from a number of different data sources, including Yahoo or MSN.

Syntax

=smfGetOptionQuotes(tickerlist, datacodelist, [headers], [source])

Parameters

tickerlist = A list of special format ticker symbols to get quotes and/or other data on options. This list can either be a single ticker symbol, a comma-delimited list of ticker symbols, a cell reference, or a range reference. The ticker symbols being used for this function are a special format of the add-in -- not ticker symbols used by the different data sources. Each ticker symbol consists of five strings with a space in between (e.g. "SPY Jun 2010 $100 Call"). The five strings are:

- Ticker symbol = The ticker symbol of the underlying equity (e.g. "MMM" or "SPY").
- Expiration Period = There are a number of choices for this string:

String	Description
Jan thru Dec	Month-end option expiration
1 thru 12	Month-end option expiration
Q1 thru Q4	Quarter-end option expiration
W or Week	Week-end option expiration
m/d	An actual month and day option expiration (e.g. 9/10)

Note that week-end and quarter-end option expirations are only available for a limited number of equities.

- Expiration Year = A 4-digit expiration year (e.g. "2010").
- Strike Price = The desired strike price, either with or without a leading dollar sign. In addition, you can use values like "OTM1", "OTM2", "ITM1", or "ITM2" to get the first or second in-the-money or out-of-the-money strike prices (currently

only available for Yahoo and MarketWatch data sources).

- Call/Put = A literal value of "Put" or "Call" to indicate the type of option data that is desired. Can be abbreviated to "P" or "C".

datacodelist = A list of data codes telling indicating which data elements you want retrieved. This list can either be a string concatenation of all desired data codes, a cell reference, or a range reference. Here is a table indicating the data codes available from the various data sources:

Code	Description	Data sources								
		(1) Y	(2) MSN	(3) OX	(4) MW	(5) OX2	--	--	--	--
%	% Change	--	Fast	--	--	--	--	--	--	--
a	Ask Price	Fast	Fast	Fast	Fast	Fast	--	--	--	--
b	Bid Price	Fast	Fast	Fast	Fast	Fast	--	--	--	--
c	$ Change	Slow	Fast	--	Fast	Fast	--	--	--	--
e	Bid Size	--	Slow	--	--	--	--	--	--	--
f	Ask Size	--	Slow	--	--	--	--	--	--	--
g	Daily Low	Slow	Slow	--	--	--	--	--	--	--
h	Daily High	Slow	Slow	--	--	--	--	--	--	--
i	Open Interest	Fast	Fast	Fast	Fast	--	--	--	--	--
j	Contract Low	Slow	--	--	--	--	--	--	--	--
k	Contract High	Slow	--	--	--	--	--	--	--	--
l	Last Price	Fast	Fast	Fast	Fast	Fast	--	--	--	--
o	Open Price	Slow	Slow	--	--	--	--	--	--	--
p	Previous Close	Slow	Slow	--	--	--	--	--	--	--
s	Strike Price	Fast	Fast	Fast	Fast	Fast	--	--	--	--
t	Last Trade Time	Slow	Slow	--	--	--	--	--	--	--
u	Underlying Equity Price	Fast	Fast	Fast	Fast	Fast	--	--	--	--
v	Volume	Fast	Fast	--	Fast	--	--	--	--	--
x	Expiration Date	Fast	Fast	Fast	Fast	Fast	--	--	--	--
y	Time/Theo. Value	--	Fast	Fast	--	--	--	--	--	--
z	Option Symbol	Fast	Fast	Fast	Fast	Fast	--	--	--	--
1	Vega	--	--	Fast	--	--	--	--	--	--
2	Theta	--	--	Fast	--	--	--	--	--	--
3	Rho	--	--	Fast	--	--	--	--	--	--
4	Gamma	--	--	Fast	--	--	--	--	--	--

5	Delta	--	--	Fast	--	Fast	--	--	--	--
6	Implied Volatility	--	--	--	--	Fast	--	--	--	--

The difference between the "Fast" and "Slow" data items is that you can retrieve all of the "Fast" items for all of the different options in the same expiration month with a single Internet access, while the "Slow" data items require a new Internet access for each individual option for which the various items are collected.

Note that all option months are retrieved by MarketWatch on a single Internet access. This can be a slower way to get the data if you only need a single expiration date, but a faster way to get the data if you need multiple expiration dates.

headers = An optional parameter that allows column headings to be generated for the returned columns of data. Defaults to 0. Set to 1 if you desire a row of headers to be displayed.

source = An optional parameter that indicates the source you want to use for options data. Defaults to "Y". Possible values:

Value	"DataCode" prefix	Data Source
MSN	2	MSN
MW	4	MarketWatch
OX	3	OptionsXPress (Greeks of calls)
OX2	5	OptionsXPress (limited Put and Call data items)
Y	1	Yahoo
2	--	Multiple sources. If this value is passed, all datacodes will be 2-byte values, where the first byte is the "DataCode" prefix shown in this table.

Examples

- This function invocation:

  =smfGetOptionQuotes("SPY Jun 2010 $100 Call,SPY Jun 2010 $110 Call","zba",1)

  ...returns:

Ticker Symbol	Bid Price	Ask Price

SPY100619C00100000	$7.84	$8.02
SPY100619C00110000	$1.17	$1.21

-
- This function invocation:

  =smfGetOptionQuotes("SPY Jun 2010 $100 Call,SPY Q2 2010 $100 Call","zba",1,"MSN")

  ...returns:

Ticker Symbol	Bid Price	Ask Price
.SPY\10F19\100.0	$7.84	$8.02
.SPY\10F30\100.0	$8.28	$8.46

-
- This function invocation, when SPY was trading at $106.82:

  =smfGetOptionQuotes("SPY Jul 2010 ITM1 Call,SPY Jul 2010 OTM1 Call","zsba",1)

  ...returns:

Ticker Symbol	Strike Price	Bid Price	Ask Price
SPY100717C00106000	$106.00	$4.63	$4.73
SPY100717C00107000	$107.00	$4.03	$4.12

-
- This function invocation:

  =smfGetOptionQuotes("SPY Aug 2010 $110 Call,SPY Aug 2010 $111 Call","1z2z3z4z",1,2)

  ...returns:

Ticker Symbol	**Ticker Symbol**	**Ticker Symbol**	**Ticker Symbol**
SPY100821C00110000	.SPY\10H21\110.0	SPY^^^100821C00110000	SPY100821C00110000

SPY100821C00111000	.SPY\10H21\111.0	SPY^^^100821C00111000	SPY100821C00111000

- ...which is basically returning the ticker symbols used by Yahoo (1z), MSN (2z), OptionsXpress (3z), and MarketWatch (4z).

- This function invocation:

  =smfGetOptionQuotes(A3:A4,B1:C1,1)

  ...returns:

	b	a
Option Symbols	Bid Price	Ask Price
SPY Jun 2010 $100 Call	$7.84	$8.02
SPY Jun 2010 $110 Call	$1.17	$1.21

- ...where the range A3:A4 containing the ticker symbols is the green-shaded area, range B1:C1 containing the data codes ot retrieve is the yellow-shaded area, and the formula was array-entered over the blue-shaded range of B2:C4.

- ...more to come

Usage notes

- In most cases, this will need to be an array-entered formula. To array-enter a formula in EXCEL, first highlight the range of cells where you would like the returned data to appear -- the number of rows for the range should be AT LEAST the number of ticker symbols you are requesting from the function, while the number of columns for the range should be AT LEAST the number of data items you are requesting for each ticker symbol from the function. Next, enter your formula and then press Ctrl-Shift-Enter.

- This function actually parses through the ticker list and data code list and does individual calls to "building block" functions like smfGetYahooOptionQuote() or smfGetMSNOptionQuote() for each data item requested.

- The "ITMn" and "OTMn" strike price options may not work properly where Yahoo intermixes the listings of the quarter-end and month-end option expirations on the same page.

- ...more to come

Documentation for smfGetYahooOptionQuote() function

Description

Returns a specific data item for an option from Yahoo.

Syntax

=smfGetYahooOptionQuote(Ticker, Put/Call, Expiry, StrikePrice, ItemID)

Parameters

Ticker = Ticker symbol of underlying equity (e.g "MMM" or "SPY").

Put/Call = A literal value of "Put" or "Call" to indicate the type of option data that is desired. Can be abbreviated to "P" or "C".

Expiry = The expiration date of the option for which data is to be retrieved. You can either pass:

- The actual option expiration date if you know it (e.g. DATE(2010,6,19)).
- The first of the month for the option expiration month you want (e.g. DATE(2010,6,1)).
- The 30th or 31st day of the month-end quarter for the option expiration quarter you want (e.g. DATE(2010,6,30)). Note that only a few index ETFs have options that expire at the end of each quarter (e.g. "SPY" or "IWM").

StrikePrice = The strike price of the option for which data is to be retrieved. You have several options here:

Actual Price	You can pass the actual strike price you want to use (e.g. 100).
ITMn	You can pass a literal to indicate you want the first in-the-money option (e.g. "ITM1"), the second in-the-money option (e.g. "ITM2"), and so forth.
OTMn	You can pass a literal to indicate you want the first out-of-the-money option (e.g. "OTM1"), the second out-of-the-money option (e.g. "OTM2"), and so forth.

ItemID = The itemID of the particular data item you want to retrieve for the option:

Faster data items	Slower data items

a	Ask Price	c	$ Change
b	Bid Price	g	Daily Low
i	Open Interest	h	Daily High
l	Last Price	j	Contract Low
s	Strike Price	k	Contract High
v	Volume	o	Open Price
x	Expiration Date	p	Previous Close
z	Actual Option Symbol	t	Last Trade Time

The difference between the "Faster" and "Slower" data items is that you can retrieve all of the "Faster" items for all of the different options in the same expiration month with a single Internet access, while the "Slower" data items require a new Internet access for each individual option for which the various items are collected.

Examples

- =smfGetYahooOptionQuote("SPY","C",DATE(2010,6,1),110,"b")
- =smfGetYahooOptionQuote("SPY","C",DATE(2010,6,19),110,"b")
- =smfGetYahooOptionQuote("SPY","C","6/19/2010",110,"b")
- ...more to come

Usage notes

- In general, you would use the smfGetOptionQuotes() function instead of this one. This is primarily intended as a "building block" function for that one.
- ...more to come

Documentation for smfGetMSNOptionQuote() function

Description

Returns a specific data item for an option from MSN.

Syntax

=smfGetMSNOptionQuote(Ticker, Put/Call, Expiry, StrikePrice, ItemID)

Parameters

Ticker = Ticker symbol of underlying equity (e.g "MMM" or "SPY").

Put/Call = A literal value of "Put" or "Call" to indicate the type of option data that is desired. Can be abbreviated to "P" or "C".

Expiry = The expiration date of the option for which data is to be retrieved. You can either pass:

- The actual option expiration date if you know it (e.g. DATE(2010,6,19)).
- The first of the month for the option expiration month you want (e.g. DATE(2010,6,1)).
- The 30th or 31st day of the month-end quarter for the option expiration quarter you want (e.g. DATE(2010,6,30)). Note that only a few index ETFs have options that expire at the end of each quarter (e.g. "SPY" or "IWM").

StrikePrice = The strike price of the option for which data is to be retrieved. You have several options here:

Actual Price	You can pass the actual strike price you want to use (e.g. 100).
ITMn	You can pass a literal to indicate you want the first in-the-money option (e.g. "ITM1"), the second in-the-money option (e.g. "ITM2"), and so forth. This option is not available at this time.
OTMn	You can pass a literal to indicate you want the first out-of-the-money option (e.g. "OTM1"), the second out-of-the-money option (e.g. "OTM2"), and so forth. This option is not available at this time.

ItemID = The itemID of the particular data item you want to retrieve for the option:

Faster data items	Slower data items

%	% Change
a	Ask Price
b	Bid Price
c	$ Change
i	Open Interest
l	Last Price
s	Strike Price
v	Volume
x	Expiration Date
y	Time Value
z	Actual Option Symbol

e	Bid Size
f	Ask Size
g	Daily Low
h	Daily High
o	Open Price
p	Previous Close
t	Last Trade Time

The difference between the "Faster" and "Slower" data items is that you can retrieve all of the "Faster" items for all of the different options in the same expiration month with a single Internet access, while the "Slower" data items require a new Internet access for each individual option for which the various items are collected.

Examples

- =smfGetMSNOptionQuote("SPY","C",DATE(2010,6,1),110,"b")
- =smfGetMSNOptionQuote("SPY","C",DATE(2010,6,19),110,"b")
- =smfGetMSNOptionQuote("SPY","C","6/19/2010",110,"b")
- ...more to come

Usage notes

- In general, you would use the smfGetOptionQuotes() function instead of this one. This is primarily intended as a "building block" function for that one.
- ...more to come

Documentation for smfGetOXOptionQuote() function

Description

Returns a specific data item for a call option from OptionsXPress.

Syntax

=smfGetOXOptionQuote(Ticker, Put/Call, Expiry, StrikePrice, ItemID)

Parameters

Ticker = Ticker symbol of underlying equity (e.g "MMM" or "SPY").

Put/Call = A literal value of "Call" or "C". Note -- only calls supported at this time. This is just to make this function's parameters consistent with the ones for MSN and Yahoo.

Expiry = The expiration date of the option for which data is to be retrieved. You can either pass:

- The actual option expiration date if you know it (e.g. DATE(2010,6,19)).
- The first of the month for the option expiration month you want (e.g. DATE(2010,6,1)).
- The 30th or 31st day of the month-end quarter for the option expiration quarter you want (e.g. DATE(2010,6,30)). Note that only a few index ETFs have options that expire at the end of each quarter (e.g. "SPY" or "IWM").

StrikePrice = The strike price of the option for which data is to be retrieved.

ItemID = The itemID of the particular data item you want to retrieve for the option:

a	Ask Price
b	Bid Price
i	Open Interest
l	Last Price
s	Strike Price
x	Expiration Date
y	Theoretical Value
z	Actual Option Symbol

1	Delta
2	Gamma
3	Rho
4	Theta
5	Vega

Examples

- =smfGetOXOptionQuote("SPY","C",DATE(2010,6,1),110,"b")
- =smfGetOXOptionQuote("SPY","C",DATE(2010,6,19),110,"b")
- =smfGetOXOptionQuote("SPY","C","6/19/2010",110,"b")
- ...more to come

Usage notes

- In general, you would use the smfGetOptionQuotes() function instead of this one. This is primarily intended as a "building block" function for that one.
- ...more to come

Documentation for smfGetOptionExpiry() function

Description

Returns the specific option expiration date for a given month and type of option.

Syntax

=smfGetOptionExpiry([Year], [Month], [Type])

Parameters

Year = An optional parameter that indicates the year for which the expiration date is to be returned. Defaults to current year.

Put/Call = An optional parameter from 1 to 12 that indicates the month for which the expiration date is to be returned. Defaults to current expiration month.

Expiry = An optional parameter that indicates the type of option expiration date to be returned. Defaults to "M". The following values can be passed:

W	Next Weekly
Week	Next Weekly
Weekly	Next Weekly
W1	Next Weekly
W2	Following Weekly
M	Monthly
Month	Monthly
Monthly	Monthly
Q	Quarterly
Qtr	Quarterly
Quarter	Quarterly
Quarterly	Quarterly

Examples

Value	**Formula (run on 2010-06-21)**
2010-06-25	=smfGetOptionExpiry(,,"W")
2010-06-25	=smfGetOptionExpiry(,,"Week")

2010-06-25	=smfGetOptionExpiry(,,"Weekly")
2010-06-25	=smfGetOptionExpiry(,,"W1")
2010-07-02	=smfGetOptionExpiry(,,"W2")
2010-07-17	=smfGetOptionExpiry(,,"M")
2010-07-17	=smfGetOptionExpiry(,,"Month")
2010-07-17	=smfGetOptionExpiry(,,"Monthly")
2010-06-30	=smfGetOptionExpiry(,,"Q")
2010-06-30	=smfGetOptionExpiry(,,"Qtr")
2010-06-30	=smfGetOptionExpiry(,,"Quarter")
2010-06-30	=smfGetOptionExpiry(,,"Quarterly")
2010-08-21	=smfGetOptionExpiry(2010,8,"M")
2010-09-18	=smfGetOptionExpiry(2010,9,"M")
2010-12-31	=smfGetOptionExpiry(2010,12,"Q")
2011-03-31	=smfGetOptionExpiry(2011,3,"Q")

Usage notes

- The "W2" option type will typically only be valid when the following week is the option month-end, because they have been creating the new weekly options over the weekend and that is the only time two weeklies will exist at the same time.
- The weekly and quarterly option expirations only exist for a few select ETFs, such as SPY and IWM.
- The weekly option types will ignore the year and month that are passed.
- ...more to come

Documentation for RCHGetYahooHistory() function

Description

Returns historical stock quotes from the Yahoo quotes interface.

Syntax

=RCHGetYahooHistory(Ticker, [Start Year], [Start Month], [Start Day], [End Year], [End Month], [End Day], [Period], [Data Items], [Header Line], [Adjust], [Resort])

Parameters

Ticker = Ticker symbol (e.g "MMM")

Start Year = Year to use as the starting date of the historical quotes; must be four digits (e.g. 2005); optional; default is to fill your array-entered range.

Start Month = Month to use as the starting date of the historical quotes; optional; default is to fill your array-entered range.

Start Day = Day to use as the starting date of the historical quotes; optional; default is to fill your array-entered range.

End Year = Year to use as the ending date of the historical quotes; must be four digits (e.g. 2005); optional; defaults to most recent date available.

End Month = Month to use as the ending date of the historical quotes; optional; defaults to most recent date available.

End Day = Day to use as the ending date of the historical quotes; optional; defaults to most recent date available.

Period = Type of data period to retrieve; optional; defaults to "d"; possible values:

"d" = Daily quotes
"w" = Weekly quotes (Yahoo chooses to use the first trading day of the week as the end of the period)
"m" = Monthly quotes (Yahoo chooses to use the first trading day of the month as the end of the period)
"v" = Dividend history ("T" is the only "Data Item" that makes a difference – Date and Dividend amount are automatic)

Data = A character string indicating which data items to retrieve; optional; defaults to

Items "DOHLCVA"; possible values within the character string:

"T" = Ticker symbol
"D" = Ending date of the period
"O" = Opening price of the period
"H" = High price for the period
"L" = Low price for the period
"C" = Closing price for the period
"V" = Trading volume for the period
"A" = Adjusted closing price for the period

Header Line = A binary value to indicate whether headers should be returned for the column(s) of data; optional; defaults to 1; possible values:

0 = Delete column headings
1 = Display column headings

Adjust = A binary value to indicate whether data should be adjusted or not; optional; defaults to 0.

0 = Do not adjust data; returns the raw data as returned from Yahoo!
1 = Adjust O/H/L/C prices based on the value of the Adjusted Close and Close amounts returned from Yahoo!

Resort = A binary value to indicate whether data should be resorted or not; optional; defaults to 0.

0 = Do not resort data; present raw data as returned from Yahoo!; newest data will be on the top rows of table.
1 = Reverse the order of data returned from Yahoo!; oldest data will be on the top rows of table.

Examples

- This would return the seven default data items for ticker symbol IBM:

  =RCHGetYahooHistory("IBM")

  If you array-entered the above formula over a 10-row by 7-column range, you would get the D/O/H/L/C/V/A data items for the latest 9 days of history.

- This would return a single item -- the adjusted closing price for ticker symbol IBM for January 3rd, 2007:

  =RCHGetYahooHistory("IBM",2007,1,3,2007,1,3,,"A",0)

- ...more to come

Usage notes

- In most cases, this will need to be an array-entered formula. To array-enter a formula in EXCEL, first highlight the range of cells where you would like the returned data to appear -- the number of rows for the range should be the number of periods of data you are requesting from the function, while the number of columns for the range should be the number of data items you are requesting for each date from the function. Next, enter your formula and then press Ctrl-Shift-Enter.

- If the function is array-entered over a range that is larger than the amount of data available from Yahoo, the excess portion of the range will be filled with blanks.

- If the function is array-entered over a range that is smaller than the amount of data returned from Yahoo, the excess Yahoo data will simply be ignored. A user-defined function can only return data to a cell or a range that it has been given access to -- in this case, by the range used when array-entering the function.

- ...more to come

Documentation for smfPricesBetween() function

Description

Summarizes historical stock quotes for a range of data from the Yahoo historical quotes interface.

Syntax

=smfPricesBetween(Ticker, Start Date, End Date, [Fields])

Parameters

Ticker = Ticker symbol (e.g "MMM"). You can also use a special value of "Header" to return a row of column headings for the returned data items.

Start Date = Date to use as the starting date for the historical quotes date range; must be an EXCEL serial date value (e.g. NOW() or DATE(2006,10,15)).

End Date = Date to use as the ending date for the historical quotes date range; must be an EXCEL serial date value (e.g. TODAY() or DATE(2007,10,15)).

Fields = Which of the 10 possible fields to return; optional; default is to return all 10 fields (i.e. "0102030405060708091O"). The 10 fields are:

01 = Open Date
02 = Open Price
03 = High Date
04 = High Price
05 = Low Date
06 = Low Price
07 = Close Date
08 = Close Price
09 = Volume
10 = Previous Close

Note that each MUST be specified as a 2-digit number. So, if all you wanted was the "Previous Close" and the "Close Price", you would ask for fields "1008".

Examples

- This would return the ten default data items for ticker symbol IBM for the past year:

  =smfPricesBetween("IBM",TODAY()-365,TODAY())

If you array-entered the above formula over a 1-row by 10-column range, you would get the O/H/L/C/V/PC data items and dates for the various items.

- If you wanted to know the 2007 YTD change for ticker "IBM", you could use:

  =smfPricesBetween("IBM",DATE(2007,1,1),DATE(2007,12,31))

  For example, using that and =smfPricesBetween("Header") on the line above it to get the headers, here's what the results looked like:

Open Date	Open Price	High Date	High Price	Low Date	Low Price	Close Date	Close Price	Volume	Previous Close
2007-01-03	$80.51	2007-10-11	$121.46	2007-07-18	$71.46	2007-10-15	$118.03	3043106100	$80.27

-
- Suppose in the above example, all I was interested in was the close prior to the starting date and the latest close of the period -- I could then have used these two formula:

  =smfPricesBetween("Header",,,"1008")
  =smfPricesBetween("IBM",DATE(2007,1,1),DATE(2007,12,31),"1008")

  ...which would have returned:

Previous Close	Close Price
$80.27	$118.03

-
- ...more to come

Usage notes

- In most cases, this will need to be an array-entered formula. To array-enter a formula in EXCEL, first highlight the range of cells where you would like the returned data to appear -- the number of rows for the range should be the number of periods of data you are requesting from the function, while the number of columns for the range should be the number of data items you are requesting for each date from the function. Next, enter your formula and then press Ctrl-Shift-Enter.

- In most cases, you will probably want to use cell references for the values. For example:

=smfPricesBetween(D17,E17,F17,"1008")

...where cell D17 contains the ticker symbol and cells E17 and F17 contain appropriate date values.

- ...more to come

Documentation for smfPricesByDates() function

Description

Downloads adjusted closing prices of a single ticker symbol for specified dates from the Yahoo historical quotes interface.

Syntax

=smfPricesByDates(Ticker, date, [date,] [date,] ...)

Parameters

Ticker = Ticker symbol (e.g "MMM").

date = A date or worksheet range containing dates for which to grab an adjusted closing price; must be an EXCEL serial date value (e.g. DATE(2006,10,15)) or a range of such date values. Strings with valid dates are also allowed (e.g. "12/31/2008"). Non-date or invalid date values will cause "#N/A" to be returned for the adjusted closing price.

Examples

- =smfPricesByDates("IBM", DATE(2007,1,1))
- =smfPricesByDates("IBM", "1/1/2007")
- =smfPricesByDates(A2, B$1:J$1)
- =smfPricesByDates(A2, B$1:J$1, DATE(2007,1,1))
- ...more to come

Usage notes

- When passing more than one date for which to retrieve data, this will need to be an array-entered formula. To array-enter a formula in EXCEL, first highlight the range of cells where you would like the returned data to appear -- the number of rows for the range should be the number of periods of data you are requesting from the function, while the number of columns for the range should be the number of data items you are requesting for each date from the function. Next, enter your formula and then press Ctrl-Shift-Enter.

- If a date you passed is not a valid trading date but is within the range of historical dates available from Yahoo, you will get the adjusted closing price from the most recent trading day.
- "#N/A" would be returned as the adjusted closing price for any date value passed that cannot be resolved -- for example, if you pass a non-date string or an empty cell, or pass a date outside of the historical range of dates available on Yahoo.
- This function is only available in version 2.0i or later of the add-in.
- ...more to come

Documentation for RCHGetHTMLTable() function

This function is used to extract an HTML table from a web page.

Syntax:

=RCHGetHTMLTable(URL, Find Begin, Begin Direction, Find End, End Direction)

…where:

URL = URL of the web page to retrieve.

Find Begin = String to search for on web page to find start of table.

Begin Direction = Number of <TABLE tags to searh for after finding the above string to find the start of the table. A negative number indicates to search backwards, positive number forwards.

Find End = String to search for on web page to find end of table. If blank, the "**Find Begin**" parameter will be reused.

End Direction = Number of </TABLE tags to searh for after finding the above string to find the end of the table. A negative number indicates to search backwards, positive number forwards.

Usage Notes:

- This function returns an array of data (the HTML table), so it needs to be array-entered. To array-enter a formula in EXCEL, first highlight the range of cells where you would like the returned data to appear -- the number of rows and columns for the range will depend on the size of the table you are retrieving and how much of that table you want to see. Next, enter your formula and then press Ctrl-Shift-Enter.

- What it does -- given this invocation:

  =RCHGetHTMLTable("http://finance.yahoo.com/q/ks?s=MMM","Market Cap (intraday)",-1,"",1)

  The function will:

  -- Retrieve HTML source of the Yahoo! Key Statistics web page.
  -- Search for "Market Cap (intraday)" within the source of the web page.
  -- Set the start of the HTML table to be the first "<TABLE" tag prior to that string (i.e. -1).
  -- Search for "Market Cap (intraday)" within the source of the web page.
  -- Set the end of the HTML table to be the first "</TABLE" tag after that string (i.e. 1).
  -- Return the full table specified by and including the found "<TABLE" and "</TABLE"

tags.

Examples:

=RCHGetHTMLTable("http://finance.yahoo.com/q/ks?s=MMM","PEG Ratio",-1,"",1)
=RCHGetHTMLTable("http://finance.yahoo.com/q?d=t&s=IBM","Volume:",-1,"",1)
=RCHGetHTMLTable("http://finance.yahoo.com/q/ao?s=IBM", "Mean Recommendation", -3, "Mean Recommendation", 1)
=RCHGetHTMLTable("http://finance.yahoo.com/q/ao?s=IBM", "Mean Target", -3, "Mean Target", 1)
=RCHGetHTMLTable("http://finance.yahoo.com/q/ao?s=IBM", "Three Months Ago", -4, "Three Months Ago", 1)
=RCHGetHTMLTable("http://finance.yahoo.com/q/ud?s=IBM", "Research Firm", -1, "Research Firm", 1)
=RCHGetHTMLTable("http://finance.yahoo.com/q/ae?s=IBM", ">Earnings Est", -2, ">Growth Est", 1)

Documentation for RCHGetTableCell() function

Description

Extracts a specified table cell from a web page.

Syntax

=RCHGetTableCell(URL, Cell#, [Find1], [Find2], [Find3], [Find4], [Row#], [EndMarker], [Look#], [ErrorMsg], [Type])

Parameters

URL = Web page to retrieve the table cell from.

Cell# = The number of cells to skip forward (after function is positioned on the page by "Find1" thru "Find4" and "Row#") before returning data.

Find1 = An optional string value to search for to position the function on the page before skipping ahead rows and cells to find the data to return. Defaults to "<BODY".

Find2 = An optional string value to search for to further position the function on the page (after finding the "Find1" string) before skipping ahead rows and cells to find the data to return. Defaults to " ".

Find3 = An optional string value to search for to further position the function on the page (after finding the "Find1" thru "Find2" strings) before skipping ahead rows and cells to find the data to return. Defaults to " ".

Find4 = An optional string value to search for to further position the function on the page (after finding the "Find1" thru "Find3" strings) before skipping ahead rows and cells to find the data to return. Defaults to " ".

Row# = An option number of rows to skip ahead (after function is positioned on the page by "Find1" thru "Find4") before skipping ahead the specified number of table cells to find the data to return. Defaults to 0.

EndMarker = An optional string value that marks the end of the skip aheads based on "Cell#" and "Row#". If the next found table cell ia after this point, the error message is returned. Defaults to "</BODY", but is usually set to "</TABLE" when using "Row#" to ensure that the search doesn't go outside the current table when skipping ahead by table rows.

Look# = An optional number of consecutive cells to search for data in (ignoring empty

table cells). Rarely used. Defaults to 0.

ErrorMsg = An optional value to return if the table cell cannot be found based on specified parameters. Defaults to "Error".

Type = An optional integer value to determine the type of Internet request to make. Defaults to 0. For now, other values are experimental:

0 = XMLHTTP "Get" Request
1 = IE Object Request
2 = HTMLDocument Request
3 = XMLHTTP "Post" Request

Examples

- To retieve the "Market Cap" value from the Yahoo! Key Statistics page for ticker symbol "MMM":

  =RCHGetTableCell("http://finance.yahoo.com/q/ks?s=MMM",1,"Market Cap (intraday)")

  ...which tells the function to retrieve the "http://finance.yahoo.com/q/ks?s=MMM" web page, look for the "Market Cap (intraday)" string on the page, then return the data in the following table cell.

- ...more to come

Usage notes

- This is the general process the function uses to extract the data:

  1. The source of the web page specified by "URL" is retrieved from the Internet.
  2. A position pointer is set to 1.
  3. The position pointer is advanced to the next location of the string specified by "Find1" found in the web page source.
  4. If "Find2" is nonblank, the position pointer is advanced to the next location of the string specified by "Find2" found in the web page source.
  5. If "Find3" is nonblank, the position pointer is advanced to the next location of the string specified by "Find3" found in the web page source.
  6. If "Find4" is nonblank, the position pointer is advanced to the next location of the string specified by "Find4" found in the web page source.
  7. If "Row#" is not zero, the ending position of the table is set by finding the string specified by "EndMarket".
  8. If "Row#" is not zero, the position pointer is advanced the number of table rows requested, to the start of the table row. If the next row found is beyond the position set by "EndMarket", an extraction error is signaled.

9. The position pointer is advanced the number of table cells specified by "Cell#". If the end of the current table row is hit before the cell is found, an extraction error is signaled.
10. If "Look#" is zero, the current cell is returned. Otherwise, it looks for and returns the first non-empty cell up to the number specified by "Look#".

- Basically, this works off the same extraction engine that the RCHGetElementNumber() function uses. The vast majority of data elements is defined by a line like:

  941;YahooKS;Market Cap;http://finance.yahoo.com/q/ks?s=~~~~~;1;VALUATION MEASURES;MARKET CAP; ; ;0;0;0;0

  That data line more or less is the same definition that this function uses. These are the relevant components:

  Element Number = 941

  Source Name (i.e. a name assigned to a specific "URL") = "YahooKS"

  Element Name = "Market Cap"

  Web Page (i.e. "URL", with "~~~~~" as the ticker symbol) = "http://finance.yahoo.com/q/ks?s=~~~~~"

  # of cells to skip ahead (i.e. "Cell#") = 1

  1st search string (i.e. "Find1") = "VALUATION MEASURES"

  2nd search string (i.e. "Find2") = "MARKET CAP"

  3rd search string (i.e. "Find3") = " "

  4th search string (i.e. "Find4") = " "

  # of rows to skip ahead (i.e. "Row#") = 0

  End of table marker (i.e. "EndMarker") = 0

  # of cells to look into (i.e. "Look#") = 0

  Type of Internet request to issue (i.e. "Type") = 0

- This function uses the same caching technique as the RCHGetElementNumber() function, so if you are retrieving multiple elements from the same page, only one web

page retrieval needs to be done. The source of the web page will be saved and used for extracton of later data elements.

- If someone does used this to define a number of elements on a page I haven't implemented, it should be a fairly trivial task for me to take their cell extraction data and convert them into a set of legitimate data elements for the RCHGetElementNumber() function.

- ...more to come

Documentation for RCHGetWebData() function

Description

Extracts source data from a web page. The primary purpose of this function is for testing, to examine the web page data being returned to VBA for processing. However, it can also be used for ad hoc extractions of data from a web page that isn't table oriented.

Syntax

=RCHGetWebData(URL, [Position], [Length], [Offset])

Parameters

URL = Web page to retrieve source data to extract from.

Position = An optional parameter that has a default value of 1, indicating either:

1. A number indicating an absolute position on the page to begin the extraction of data
2. A string to search for on the page to indicate a relative position on the page to begin the extraction of data

Length = An optional parameter that has a default value of 32767 (the maximum possible value), indicating the length of the data to extract from the web page.

Offset = An optional parameter that has a default value of 0, indicating the relative position to offset from parameter "Position" for extraction of data from the web page.

Examples

- To retieve the first 32767 bytes of the Yahoo! Key Statistics page for ticker symbol "IHR":

  =RCHGetWebData("http://finance.yahoo.com/q/ks?s=IHR")

- To retrieve successive data from the same web page, you could use these additional formulas:

  =RCHGetWebData("http://finance.yahoo.com/q/ks?s=IHR",32768,32767)

  =RCHGetWebData("http://finance.yahoo.com/q/ks?s=IHR",65535,32767)

- ...more to come

Usage notes

- A value of "Error" is returned if the "Position" or "Length" values are invalid for the web page. However, if the "Position" is within the web page and the "Length" would cause the extraction to go outside of the web page, the "Length" is reset so the extraction only goes to the length of the web page.

- ...more to come

Documentation for smfGetCSVFile() function

Description

Parses data from a CSV file into an EXCEL range.

Syntax

=smfGetCSVFile(URL)

Parameters

URL = CSV file to parse data from (can also be a local file name instead of a URL).

Examples

- To retrieve NASDAQ advancing issues data from Unicorn:

  =smfGetCSVFile("http://unicorn.us.com/advdec/NASDAQ_advn.csv")

- To retrieve a local file:

  =smfGetCSVFile("C:\Users\Randy\Documents\Temp\Test3.csv")

- An alternative way to grab current quotes data from Yahoo:

  =smfGetCSVFile("http://download.finance.yahoo.com/d/quotes.csv?s=MMM&f=sl1d1t1c1ohgv&e=.csv")

- An alternative way to grab dividend data from Yahoo historical quotes:

  =smfGetCSVFile("http://ichart.finance.yahoo.com/table.csv?s=MMM&g=v&ignore=.csv")

- ...more to come

Usage notes

- This function returns an array of data, so it needs to be array-entered. To array-enter a formula in EXCEL, first highlight the range of cells where you would like the returned data to appear -- the number of rows and columns for the range will depend on the size of the CSV file you want parsed and how much of that CSV file you want to see. Next, enter your formula and then press Ctrl-Shift-Enter.

- This function is only available in version 2.0i or later of the add-in.

- ...more to come

Documentation for smfGetEconData() function

Description

Extracts economic data from the St. Louis Federal Reserve Economic Research data pages (i.e. FRED).

Syntax

=smfGetEconData(SeriesID, Date)

Parameters

SeriesID = The "series identifier" used by FRED to identify the different series of data they have available. For example:

DGS7 = Daily 7-Year Treasury Constant Maturity Rate
ICSA = Weekly Initial Claims of Unemployment, Seasonally Adjusted

There are hundreds of data series avaialable: FRED Home Page

Date = Date for which to extract data for; must be an EXCEL serial date value (e.g. NOW() or DATE(2006,10,15)). In addition, labels can be sent in the date field to retrieve specific header items from the file:

Label
Title
Series ID
Source
Release
Seasonal Adjustment
Frequency
Units
Date Range
Last Updated
Notes

Examples

- To retrieve the daily value of the 7-Year Treasury Constant Maturity Rate for 1/2/2008:

  =smfGetEconData("DGS7",DATE(2008,1,2))
- To retrieve the source for the 7-Year Treasury Constant Maturity Rate:

  =smfGetEconData("DGS7","Source")
- ...more to come

Usage notes

- None at this time.
- ...more to come

Documentation for smfGetTagContent() function

Description

An experimental (i.e. it might change) function that extracts the content from a specified HTML tag from the source code of a web page.

Syntax

=smfGetTagContent(URL, Tag, Tag#, [Find1], [Find2], [Find3], [Find4], [Convert], [ErrorMsg], [Type])

Parameters

URL = Web page to retrieve the HTML tag content from.

Tag = The type of tag to retrieve content from -- "table", "td", "li", "span", etc.

Tag# = The number of tags to skip backward or forward, after function is positioned on the page by "Find1" thru "Find4", before returning data.

Find1 = An optional string value to search for to position the function on the page before skipping backward or forward to find the data to return. Defaults to "<", which should position you at the top of the document.

Find2 = An optional string value to search for to further position the function on the page (after finding the "Find1" string) before skipping backword or forward to find the data to return. Defaults to " ".

Find3 = An optional string value to search for to further position the function on the page (after finding the "Find1" thru "Find2" strings) before skipping backward or forward to find the data to return. Defaults to " ".

Find4 = An optional string value to search for to further position the function on the page (after finding the "Find1" thru "Find3" strings) before skipping backward or forward to find the data to return. Defaults to " ".

Convert = An indicator of whether the data found within the tag should be converted or parsed. Unused at this time. Defaults to 0.

ErrorMsg = An optional value to return if the table cell cannot be found based on specified parameters. Defaults to "Error".

Type = An optional integer value to determine the type of Internet request to make.

Defaults to 0. For now, other values are experimental:

0 = XMLHTTP "Get" Request
1 = IE Object Request
2 = HTMLDocument Request
3 = XMLHTTP "Post" Request

Examples

- To retieve the "Title" value from the Yahoo! Key Statistics page for ticker symbol "MMM":

  =smfGetTagContent("http://finance.yahoo.com/q/ks?s=MMM","title",1)

  ...which tells the function to retrieve the "http://finance.yahoo.com/q/ks?s=MMM" web page, look for the first "<title>" HTML tag on the page, then return the data within that HTML tag. In this case, it returns "MMM: Key Statistics for 3M Company Common Stock - Yahoo! Finance".

- ...more to come

Usage notes

- This function uses the same caching technique as the RCHGetElementNumber() function, so if you are retrieving multiple elements from the same page, only one web page retrieval is done by the add-in. The source of the web page will be saved and used for extraction of later data elements.

- ...more to come

Documentation for RCHCreateComment() function

Description

Creates a comment box for the cell that can contain text and/or an image (e.g. a stock chart).

Syntax

=RCHCreateComment(Ticker, [Choice], [Width], [Height], [Visible], [Top], [Left], [Scale], [Text], [Return])

Parameters

Ticker = Ticker symbol (e.g "MMM")

Choice = Type of date to insert into created comment. Defaults to a value of 1. Possible values:

0 = Insert text only
1 = Insert image of daily chart from Stockcharts' Gallery View for passed "Ticker"
2 = Insert image of P&F Chart from StockCharts for passed "Ticker"
3 = Insert image of 6-month Candleglance Chart from StockCharts for passed "Ticker"
4 = Insert image of 6-month chart from Business Week Online for passed "Ticker"
99 = Insert custom image using the "Ticker" field value as the URL of the image

Width = Width (in pixels) of comment box. Optional except when Choice=99. Default values vary depending on the value of "Choice".

Height = Height (in pixels) of comment box. Optional except when Choice=99. Default values vary depending on the value of "Choice".

Visible = A binary value to indicate whether the comment should be visible or hidden; optional; defaults to 0 (hidden).

0 = Keep comment hidden
1 = Make comment visible

Top = Top position (in pixels) of comment relative to the top of the cell containing the formula. Defaults to 1.

Left = Leftmost position (in pixels) of comment relative to the left of the cell containing the formula. Defaults to 1.

Scale = Scaling amount to apply to Choice-based height and width sizes. Defaults to 100%.

Text = Text data to place into comment box. In most cases, you would NOT use this if you are loading an image. Defaults to "".

Return = Data to return as the value for the cell. Defaults to "Chart".

Examples

- Either of these would insert a comment box containing the daily chart from Stockcharts' Gallery View for ticker "IHR":

  =RCHCreateComment("IHR",1,0,0,1,2,1,1,"","Chart")

  =RCHCreateComment("IHR",1)

- This would insert a comment box containing a custom chart for ticker "HLX":

  =RCHCreateComment("http://stockcharts.com/c-sc/sc?s=HLX&p=D&yr=0&mn=6&dy=0&i=t39628903145&r=9933",99,350,360)

- ...more to come

Usage notes

- ...none at this time

- ...none at this time

Documentation for smfTech() function

Description

BETA version Allows the creation of technical analysis indicators from a range of D/O/H/L/C/V data.

Syntax

=smfTech(datarange, indicator, [parm1], [parm2], [parm3], [parm4], [parm5], [parm6])

Parameters

datarange = A range or array of data consisting of 7 columns of data (Date, Open, High, Low, Close, and Volume), sorted in ascending date sequence with the first row as text headers.

indicator = A string value indicating the type of technical analysis values to create:

```
    ADL  = Accumulation/Distribution Line
           -- No parameters are used

    ATR  = Average True Range
           -- parm1 = period, defaults to 20

    CCI  = Commodity Channel Index
           -- parm1 = period, defaults to 20

    EMA  = Exponential Moving Average
           -- parm1 = period, defaults to 50

    MACD = Moving Average Convergence Divergence
           -- parm1 = period of 1st moving average, defaults to 12
           -- parm2 = period of 2nd moving average, defaults to 26
           -- parm3 = smoothing period for difference of the two
moving averages, defaults to 9

    OBV  = On Balance Volume
           -- No parameters are used

    ROC  = % Rate of Change
           -- parm1 = period, defaults to 21

    RSI  = Relative Strength Index
           -- parm1 = period, defaults to 20

    SMA  = Simple Moving Average
           -- parm1 = period, defaults to 50

    STO  = Stochastics
           -- parm1 = %K period, defaults to 14
```

```
-- parm2 = %D period, defaults to 5
-- parm3 = smoothing period, defaults to 1
```

parm1-parm6 = Parameters to be used to create the technical analysis values. See the "indicator" list for details on each.

Examples

- See the template in the files area for sample usage of each.

Usage notes

- Verify values before trusting...this is a function that is under development.
- This function needs to be array-entered when using it to return a range of data in the workbook. In some cases, an indicator may return multiple columns of data. However, the first column contains the values for the indicator. If they exist, the other columns will contain intermediate calculations that may be of use.
- This function can also be used to generate single "on the fly" technical analysis values. For example:

  =INDEX(smfTech(RCHGetYahooHistory("MMM",,,,,,,,,1,1,11,6),"SMA",10),11)

  ...would return the current 10-day simple moving average for ticker "MMM". That is:

  - RCHGetYahooHistory("MMM",,,,,,,,,1,1,11,6)

    ...gets 10 days of historical quotes (plus the header row).

  - smfTech(...,"SMA",10)

    ...generates the 10-day simple moving average on that returned historical data.

  - INDEX(...,11)

    ...gets the 11th array element returned -- which is the current day's value. That's because the Yahoo historical quotes are sorted from oldest data to newest data.

- ...more to come

Documentation for smfStrExtr() function

Description

A utility function to extract a substring from another string.

Syntax

=smfStrExtr(Haystack, Start, End)

Parameters

Haystack = String to extract the substring from.

Start = String to start the extract of the substring from, FOLLOWING this string when found.

End = String to end the extract of the substring from, BEFORE this string when found. The search for this string will be after the ending of the "Start" string

Examples

- To extract "table" out of "1table2", you would use:

  =smfStrExtr("1table2","1","2")

  ...which says to extract everything between "1" and "2".

- To extract "table" out of "1table12", you would use:

  =smfStrExtr("1table12","1","1")

  ...which says to extract everything between the first "1" and the "1" following that.

- ...more to come?

Usage notes

- None at this time...
- ...more to come?

Stock Market Functions Add-In Change Log

Date	Release	Description	Element Numbers
2007/01/20	1.3h	• Change RCHGetYahooQuotes() to work with the new downloader URL that Yahoo! is using	---
		• Changed RCHGetYahooHistory() to work with the new date defaults that Yahoo! is using	---
		• Changed all numeric conversion to CDec() from CCur() to increase precision	---
		• Fix Earnings.com extract of dividends and splits for situations when no splits have ever occurred	1401-1512
		• Fix TickerReset() and ArrayReset() processing to initialize arrays properly	---
		• Fix RCHGetTableCell() to return vError instead of blanks when not finding new rows	---
		• Fix conversion of "- " and "-- " to return zero values	---
		• Fix ADL indicator of SMFTech() function to account for days when High and Low price are the same	---
2006/12/22	1.3g	• Added the ability to use a server prefix with RCHGetYahooQuotes() to get quotes using Yahoo Quotes servers set up for other countries	---
		• Removed MsgBox with error message upon failure to access the Internet because of problems it caused with automated routines	---
		• Add SMFForceRecalculation() macro to aid in automated recalculation of all data	---
		• Fix processing to convert M/B into Million and Billions on non-numeric data	---
		• Fix several MSN tags related to 5-year average growth rates (Canadian stocks had different tag)	---
		• Change all HTML table header tags to normal table cell tags on web pages retrieved to aid extraction from improperly coded web pages	---
		• Fix RCHGetYahooHistory() date validation, using 2-digit months and days	---
		• Add SMFTech() function, to create technical analysis arrays from D/O/H/L/C/V historical data. A template is here.	---
		• Add smfUpdateDownloadTable() macro, to allow	---

		downloading of a table of data. A sample is here.	
2006/09/12	1.3f	• Fixed RCHGetYahooQuotes() parsing -- non-double-quoted fields within comma-delimited data were dropping the last character	---
2006/09/10	1.3e	• Fixed RCHGetYahooQuotes() parsing to handle double-quoted fields within comma-delimited data	---
		• Changed web page source of MSN 10-year summary of financial statement items (MSN page reformat)	243-352
		• Obsoleted MSN financial statement data elements (MSN page reformat to use Reuters data)	1516-1540 1542-1551 3974-4923
		• Added RCHCreateComment() function	---
		• Added ability of RCHGetHTMLTable() function to extract rows from a table	---
		• Added RCHGetTableCell() function	---
		• Added RCHGetWebData() function	---
2006/08/17	1.3d	• Added sector and industry names/numbers from Yahoo	13864-13868
		• Fixed a number of elements from Zacks due to web page changes	848-876
		• Added a number of elements from Zacks due to web page changes	13869-13890
		• Added annual O/H/L/C prices from Business Week (values appear inaccurate in some cases)	13891-13930
		• Added a number of Valuation Ratio elements from Morningstar	13931-14067
		• Added a number of Key Ratio elements from Morningstar	14058-14398
		• Added the currency type and magnitude for Google Financial Statements data	14399-14400
		• Added a number of calculated fields (e.g. estimates of Piotroski, Altman, MFI, Rule #1)	15001-15014
		• =RCHGetElementNumber("Web Page", elementnumber) will return the URL of the web page the data element is retrieved from	---
		• Removed volume adjustment option from =RCHGetYahooHistory(). Apparently, all volume is ALWAYS provided adjusted for stock splits.	---
2006/07/18	1.3c	• Added a number of Comparison Ratio elements from	13626-

		Reuters	13821
		• Added a number of Market Statistics elements from Yahoo	13822-13861
		• Added company name from either MSN or Yahoo	13862-13863
		• Changed "P&F -- Trend" (i.e. "Bearish" or "Bullish") from StockCharts to be based on price direction instead of price objective prefix	1515
		• Changed RCHGetElementNumber() function to allow for a custom error value to be returned	---
		• Fixed RCHGetHTMLTable() function to allow for existence of <THEAD></THEAD> table rows and empty table cells	---
		• Fixed RCHGetYahooHistory() function for incorrect edit of weekly data series request	---
		• Changed RCHGetYahooHistory() to allow for ascending date sorting and automatic adjustment of amounts based on the "Adjusted Close" value	---
2006/07/01	1.3b	• Added a number of mutual fund data elements from Yahoo Finance	4930-5195
		• Added 10 years of annual financial statement data from AdvFN	5196-8005
		• Added 5 years of quarterly financial statement data from AdvFN	8006-13625