TRADING
TECHNOLOGIES

X_TRADER API
RTD TUTORIAL

VERSION 7.X
DOCUMENT VERSION 7.X.DV2 3/5/14

LEGAL

Table of Contents

This page intentionally left blank for printing purposes

1 Getting Started

Chapter overview

This chapter introduces the X_TRADER® API RTD Server and shows how it fits into a typical X_TRADER® environment. It also provides an overview of the tutorial and the tutorial application.

In this chapter

About the X_TRADER® API RTD (Real Time Data) server

First came Microsoft Excel technology for accessing external data

Microsoft Excel is a spreadsheet application that allows users to enter their own values and to construct formulas to manipulate that data. Prior to Excel version 2002, linking to external data required you to use technologies known as OLE or DDE. While these technologies allowed a user to display live data updates in Excel cells, the technologies were not created for the purpose of streaming large amounts of real-time data to Excel and were plagued with inefficiencies when used in this way. Starting with Excel 2002, Microsoft introduced a new way to view real time data, called RTD (Real Time Data).

Then came the X_TRADER® API

Trading Technologies created an RTD Server that exposes much of the functionality of the TT X_TRADER® API (XTAPI). With the X_TRADER® API, traders can use Excel as a front-end screen for monitoring market data, positions, and P&L.

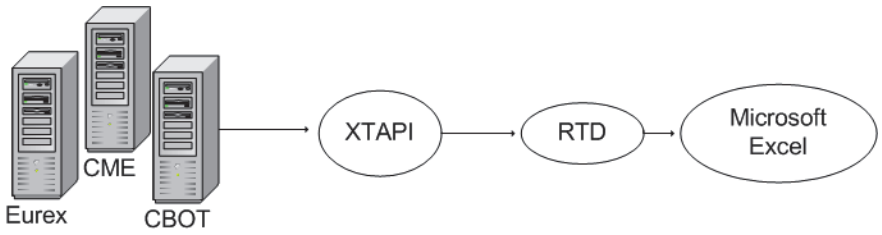
Trading Technologies distributes the X_TRADER® API as part of the installation of TT_TRADER and X_TRADER®. To use the TT XTAPI RTD Server, you must have a license for at least one of these products, as well as a valid installation of Microsoft Excel 2002 or above.

Merging the capabilities of RTD and the XTAPI

XTAPI allows customers of Trading Technologies to leverage the functionality of TT's core network technology by building their own custom applications. Software applications that use XTAPI can access live prices from every Exchange supported by TT, and can:

- Access live prices from every exchange supported by TT Gateways
- Enter and modify orders
- Receive order acknowledgements and fills
- Calculate accurate P&L

The TT RTD Server also uses the XTAPI to allow traders to display live data in an Excel spreadsheet.



Tutorial overview

Prerequisites

To complete this tutorial, you must meet the following prerequisites:

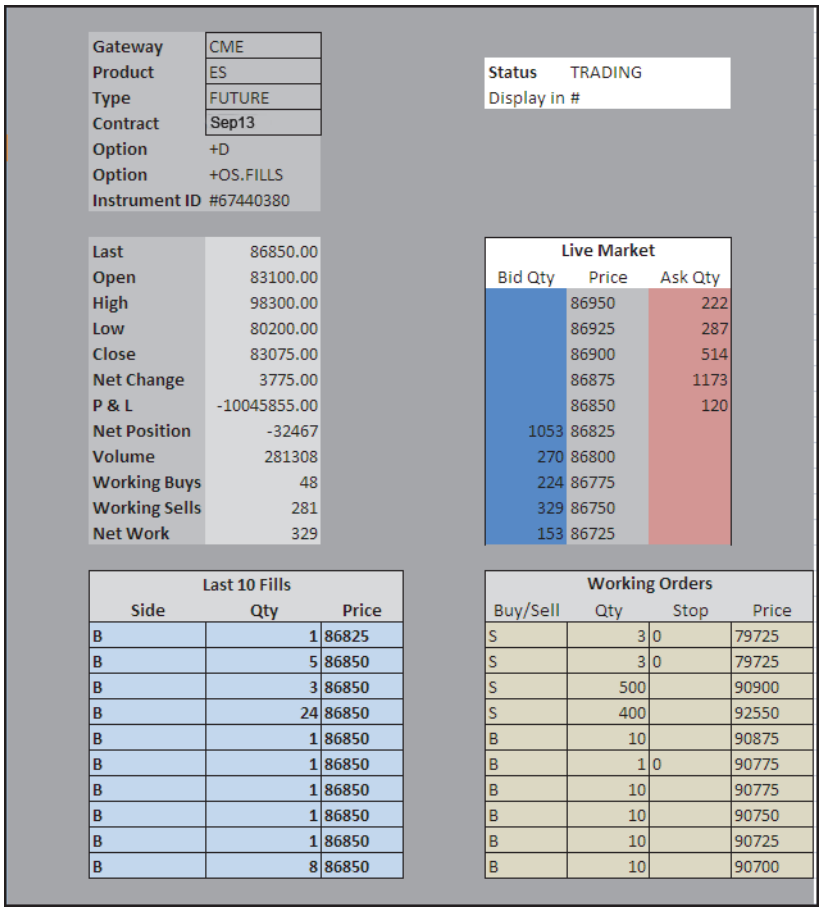
- The X_TRADER® application must be installed on your workstation.
- You must be able to access at least one TT Gateway with valid credentials.
- Your computer must have Microsoft Excel 2002 or newer. This tutorial uses Microsoft Excel 2007.

Note: This tutorial uses the TT CME Gateway and contracts available on it. If you cannot access the TT CME Gateway in your environment, you need to substitute contract information appropriate for the TT Gateway you can access throughout this tutorial.

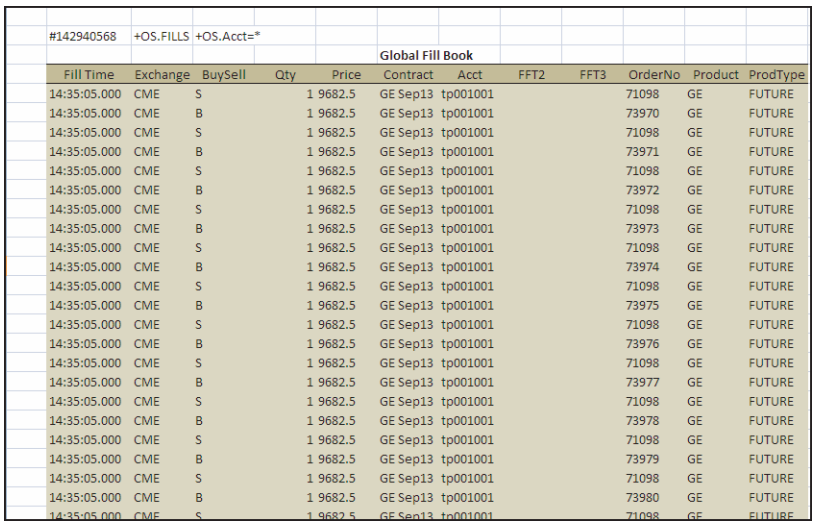
About the tutorial spreadsheet

This tutorial walks you through the process of creating two mini-applications in a single Excel spreadsheet. Each application uses a different worksheet within the spreadsheet, as follows:

- Market Monitor, which monitors market data, orders, fills for a single contract



- Global Fill Book, which displays fills for multiple contracts and accounts

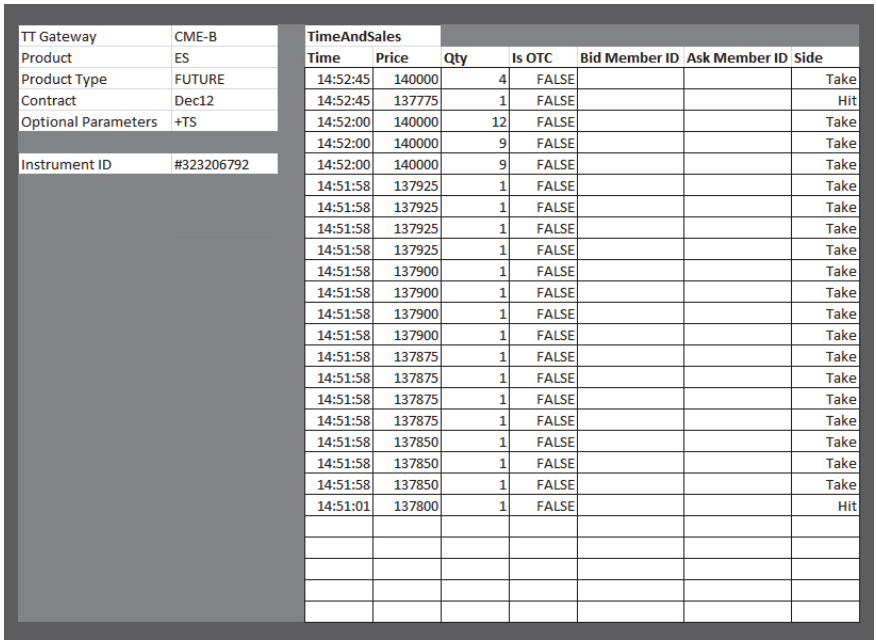


#142940568 +OS.FILLS +OS.Acct=*

Global Fill Book

Fill Time	Exchange	BuySell	Qty	Price	Contract	Acct	FFT2	FFT3	OrderNo	Product	ProdType
14:35:05.000	CME	S	1	9682.5	GE Sep13	tp001001			71098	GE	FUTURE
14:35:05.000	CME	B	1	9682.5	GE Sep13	tp001001			73970	GE	FUTURE
14:35:05.000	CME	S	1	9682.5	GE Sep13	tp001001			71098	GE	FUTURE
14:35:05.000	CME	B	1	9682.5	GE Sep13	tp001001			73971	GE	FUTURE
14:35:05.000	CME	S	1	9682.5	GE Sep13	tp001001			71098	GE	FUTURE
14:35:05.000	CME	B	1	9682.5	GE Sep13	tp001001			73972	GE	FUTURE
14:35:05.000	CME	S	1	9682.5	GE Sep13	tp001001			71098	GE	FUTURE
14:35:05.000	CME	B	1	9682.5	GE Sep13	tp001001			73973	GE	FUTURE
14:35:05.000	CME	S	1	9682.5	GE Sep13	tp001001			71098	GE	FUTURE
14:35:05.000	CME	B	1	9682.5	GE Sep13	tp001001			73974	GE	FUTURE
14:35:05.000	CME	S	1	9682.5	GE Sep13	tp001001			71098	GE	FUTURE
14:35:05.000	CME	B	1	9682.5	GE Sep13	tp001001			73975	GE	FUTURE
14:35:05.000	CME	S	1	9682.5	GE Sep13	tp001001			71098	GE	FUTURE
14:35:05.000	CME	B	1	9682.5	GE Sep13	tp001001			73976	GE	FUTURE
14:35:05.000	CME	S	1	9682.5	GE Sep13	tp001001			71098	GE	FUTURE
14:35:05.000	CME	B	1	9682.5	GE Sep13	tp001001			73977	GE	FUTURE
14:35:05.000	CME	S	1	9682.5	GE Sep13	tp001001			71098	GE	FUTURE
14:35:05.000	CME	B	1	9682.5	GE Sep13	tp001001			73978	GE	FUTURE
14:35:05.000	CME	S	1	9682.5	GE Sep13	tp001001			71098	GE	FUTURE
14:35:05.000	CME	B	1	9682.5	GE Sep13	tp001001			73979	GE	FUTURE
14:35:05.000	CME	S	1	9682.5	GE Sep13	tp001001			71098	GE	FUTURE
14:35:05.000	CME	B	1	9682.5	GE Sep13	tp001001			73980	GE	FUTURE
14:35:05.000	CME	S	1	9682.5	GE Sep13	tp001001			71098	GE	FUTURE

- Time&Sales, which displays time and sales information for a single contract (requires X_TRADER® API 7.7.8 or higher)



TT Gateway	CME-B
Product	ES
Product Type	FUTURE
Contract	Dec12
Optional Parameters	+TS
Instrument ID	#323206792

TimeAndSales

Time	Price	Qty	Is OTC	Bid Member ID	Ask Member ID	Side
14:52:45	140000	4	FALSE			Take
14:52:45	137775	1	FALSE			Hit
14:52:00	140000	12	FALSE			Take
14:52:00	140000	9	FALSE			Take
14:52:00	140000	9	FALSE			Take
14:51:58	137925	1	FALSE			Take
14:51:58	137925	1	FALSE			Take
14:51:58	137925	1	FALSE			Take
14:51:58	137925	1	FALSE			Take
14:51:58	137900	1	FALSE			Take
14:51:58	137900	1	FALSE			Take
14:51:58	137900	1	FALSE			Take
14:51:58	137900	1	FALSE			Take
14:51:58	137875	1	FALSE			Take
14:51:58	137875	1	FALSE			Take
14:51:58	137875	1	FALSE			Take
14:51:58	137875	1	FALSE			Take
14:51:58	137850	1	FALSE			Take
14:51:58	137850	1	FALSE			Take
14:51:58	137850	1	FALSE			Take
14:51:01	137800	1	FALSE			Hit

Creating the spreadsheet through this tutorial

As you progress through this tutorial and complete the exercises, you learn how to:

- Use the `RTD()` function in the Excel application.
- Create a basic RTD application template in Excel.
- Retrieve live market data for an instrument.
- Display market depth for an instrument.
- Calculate P&L.
- Display time and sales information.

2 Entering RTD Formulas

Chapter overview

This chapter introduces the formula you use in Excel to retrieve data from XTAPI. It also shows you some basic examples of the formula.

In this chapter

About the RTD() formula in Excel

What it is and what it does

Microsoft Excel provides the `RTD()` function that the XTAPI RTD Server uses to send requests to TT Gateways for information. Using the `RTD()` function, you can retrieve live, up-to-date information about:

- Instruments
- Market data
- Orders
- Fills

Structure of an RTD() formula

The TT implementation of the standard Excel `RTD()` formula uses the following basic structure.

```
=RTD("xtapi.rtd", ServerName, ObjectSpecsOrID, [AdditionalArgs])
```

where:

- *`ServerName`* represents the name of the RTD Server. For the TT XTAPI RTD Server, you must omit this parameter.
- *`ObjectSpecsOrID`* identifies which type of XTAPI object you want to query. You can specify the value as a:
  - Comma-separated list of values that identify the object (such as, **Exchange**, **Product**, **Type**, and **Contract** for an instrument)
  - Unique internal TT identifier generated by a formula for the object
- *`AdditionalArgs`* contains a comma-separated list of parameters that provide additional query details or that instruct Excel how to display array data.

This tutorial explains the `RTD()` formula and these generic parameters in more detail as you use them to accomplish specific tasks.

For more information about the `RTD()` formula, refer to the *XTAPI RTD Feature Guide*.

Entering RTD formulas

Retrieving an instrument using contract qualifiers

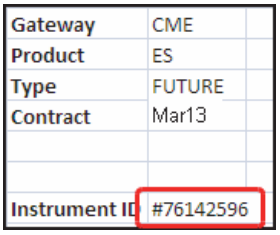
To begin the process of accessing market data, you simply tell Excel which instrument it should monitor by entering an `RTD()` formula in a cell. If, for example, you want to monitor the Mar13 ES futures contract from the TT CME Gateway, you need only enter the following formula in a cell.

```
=RTD("xtapi.rtd",,"Instr","CME","ES","FUTURE","Mar13")
```

With this formula, you instruct Excel to access the XTAPI RTD and to retrieve the instrument with the following contract qualifiers:

- Gateway = **CME**
- Product = **ES**
- Product type = **FUTURE**
- Contract = **Sep13**

After processing the formula, Excel displays the result in the cell. In this case, Excel returned a long value that represents a generated Instrument ID it uses to identify the instrument with the specified contract qualifiers, similar to the following:



Retrieving an instrument using the Instrument ID

Because you stored the formula in a cell, you can now use the result shown in the cell to identify the instrument in other formulas in the spreadsheet, instead of specifying all of the contract qualifiers each time. Assuming you entered the above formula in cell **A1**, you can reference the same instrument by entering the following formula in a different cell, such as **B1**, to show the opening price.

```
=RTD("xtapi.rtd",,A1,"Open")
```

Note: The XTAPI RTD Server generates the Instrument ID each time you open the spreadsheet or when you change the formula. Consequently, you cannot rely on the value being the same each time you open the spreadsheet. For example, in this example, you should not use **#76142596** to identify the instrument in other formulas.

So, if you wanted to know the last traded price of the instrument, you would not ask, "What is the last price of instrument **#76142596**?" Rather, you would ask, "What is the last traded price of the instrument defined in cell **A1**, where **A1** contains the formula for the Instrument ID?"

Retrieving market data for an instrument

Now that you know how to retrieve an instrument through the XTAPI RTD Server, you can start creating formulas that return live market data from the TT Gateway. To access market data, you supply the attribute of the instrument that you want to retrieve. For example, if you want to get the Last Traded Price (LTP) for an instrument, you request the **Last** attribute.

As you might surmise, you can use either type of formula to get the data, as shown in the following example.

Example: Retrieving market data for an instrument

To use the contract qualifiers, use the following formula:

```
=RTD("xtapi.rtd",,"CME", "ES", "FUTURE", "Dec13", "Last")
```

To use the Instrument ID in cell **A1**, use the following formula:

```
=RTD("xtapi.rtd",,A1, "Last")
```

Both formulas create a dynamic link between the cell in the Excel spreadsheet and the instrument attribute value in the XTAPI. As the market moves and the LTP changes, the XTAPI RTD Server automatically updates the value displayed in the cell.

Tip: You can also generate basic RTD formulas from an X_TRADER® Market Grid window and paste them into an XTAPI RTD Server spreadsheet. For more information, refer to the XTAPI RTD Server help system.

Recommended approach

By specifying the contract qualifiers once and storing the result in a cell, you can easily link different parts of a spreadsheet to a single instrument and allow users to change instruments easily. You could create a single RTD template that works for any contract on any TT Gateway. A trader could then open the spreadsheet and enter the contract qualifiers, and all of the market data automatically updates to reflect the new instrument.

Tip: While both formulas produce the same result, TT recommends that you use the generated Instrument ID in formulas because it increases flexibility and reduces the chances for referential errors.

3 Creating the RTD Template

Chapter overview

This chapter shows you how to start creating your RTD application. It shows how to set up the spreadsheet and start entering contract information. Then it shows you how to retrieve and display market data for that instrument.

In this chapter

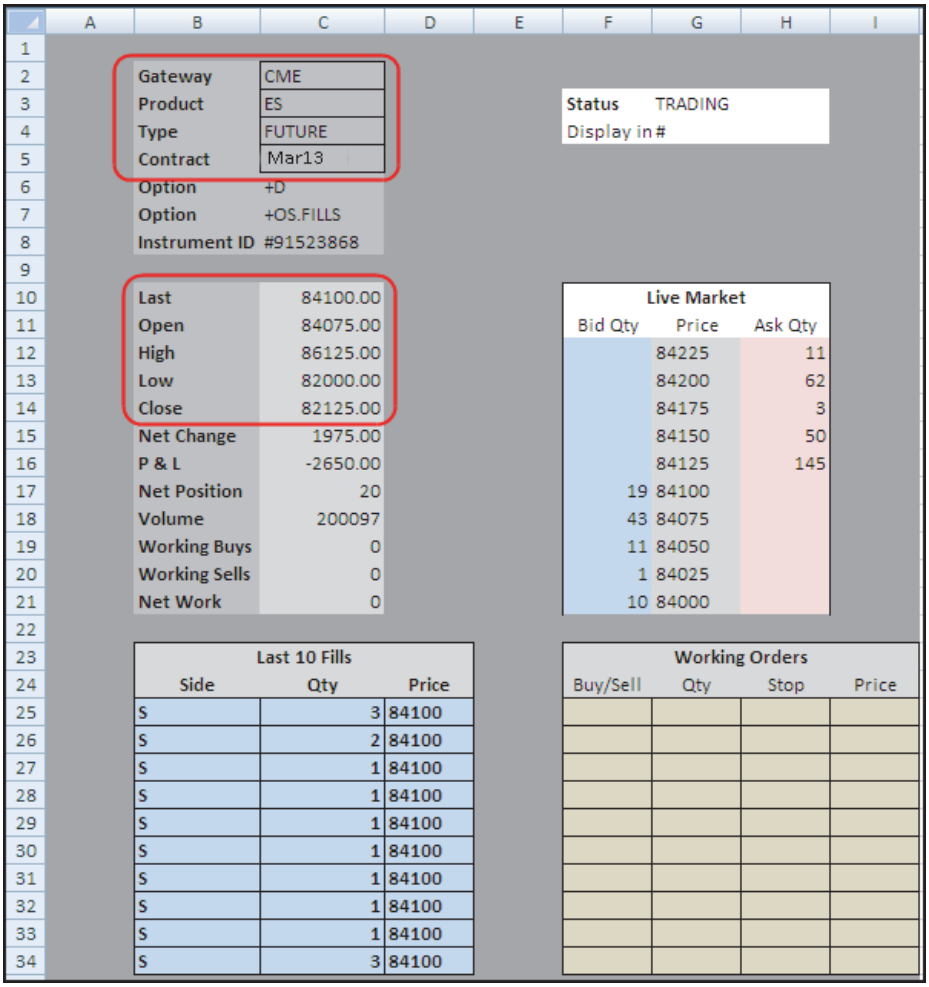
Overview

In review

In the last chapter, you learned how Microsoft Excel and the XTAPI RTD Server communicate with each other through the Excel `RTD()` function. You also saw how you could create queries by specifying contract attributes as parameters to the function.

Retrieving price data for a contract

In this chapter, you start building an RTD application by creating a spreadsheet and linking it to the XTAPI RTD Server. You then identify cells in which you can enter contract identifiers and retrieve an instrument from a TT Gateway. With the instrument, you retrieve price information, as follows.



Gateway	CME
Product	ES
Type	FUTURE
Contract	Mar13
Option	+D
Option	+OS.FILLS
Instrument ID	#91523868

Status TRADING
Display in #

Last	84100.00
Open	84075.00
High	86125.00
Low	82000.00
Close	82125.00
Net Change	1975.00
P & L	-2650.00
Net Position	20
Volume	200097
Working Buys	0
Working Sells	0
Net Work	0

Live Market

Bid Qty	Price	Ask Qty
	84225	11
	84200	62
	84175	3
	84150	50
	84125	145
19	84100	
43	84075	
11	84050	
1	84025	
10	84000	

Last 10 Fills

Side	Qty	Price
S	3	84100
S	2	84100
S	1	84100
S	1	84100
S	1	84100
S	1	84100
S	1	84100
S	1	84100
S	1	84100
S	3	84100

Working Orders

Buy/Sell	Qty	Stop	Price

Setting up the spreadsheet

Overview

To start building your RTD application, you need to:

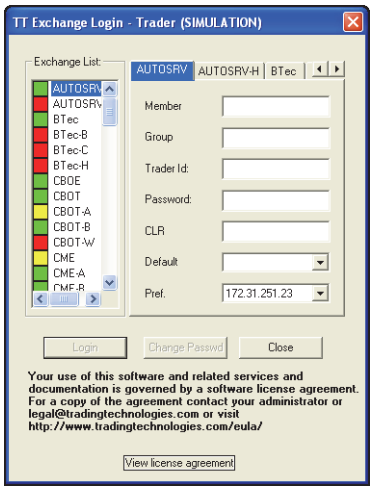
- Start X_TRADER® and connect to the TT Gateways containing the contracts you want to monitor.
- Open an Excel spreadsheet.

Connecting to a TT Gateway

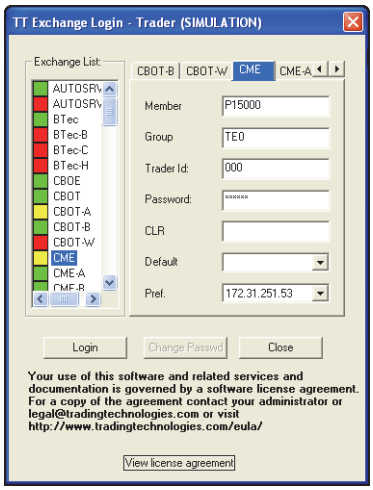
To connect to the TT CME Gateway:

1. On the desktop, double-click the X_TRADER® icon ( ) to start X_TRADER®.

   The **TT Exchange Login** dialog appears.

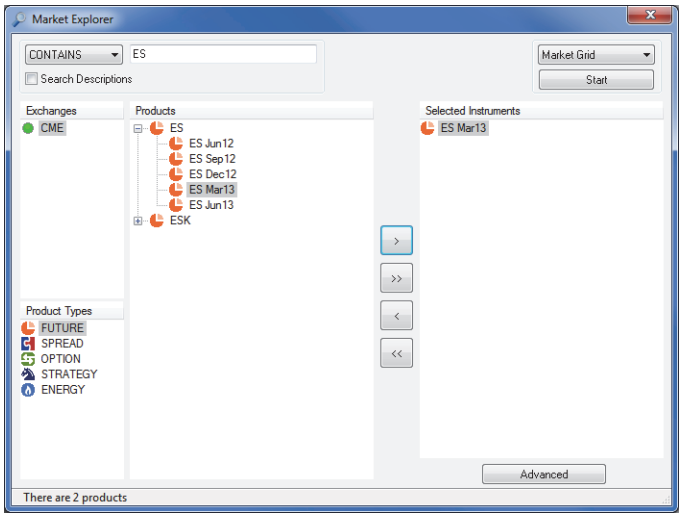


2. Login to the appropriate TT Gateway. This tutorial assumes you log into the TT CME Gateway, as shown.



3. Click **Close**.

4. From the X_TRADER® toolbar, open the **Market Explorer** window and select an instrument to trade. The rest of this chapter assumes you select **ES Mar13**.



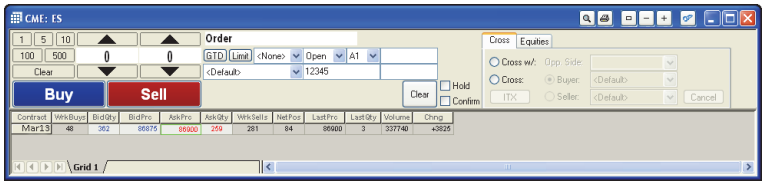
Note: You do not have to open the contract in X_TRADER® to use the RTD, but you can use the

Market Explorer

window to make sure you are working with a valid tradable instrument.

5. Click **Start**.

   The instrument opens in the X_TRADER® **Market** window.



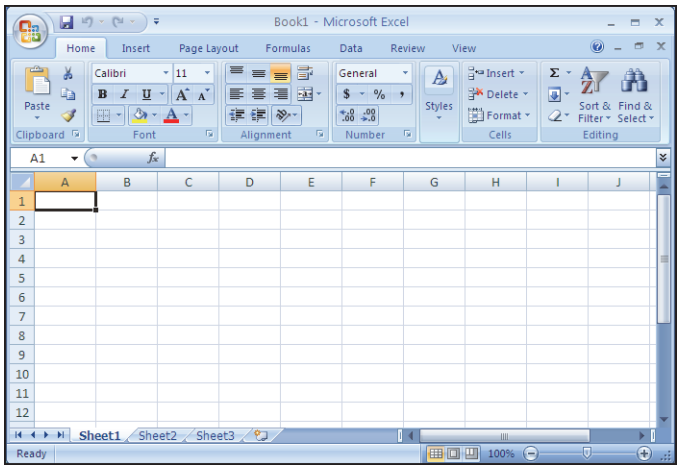
6. Close the **Market Explorer**, and minimize the **Market** window.

Opening the spreadsheet

To open the Excel spreadsheet:

1. Open the Microsoft Excel application.

   A blank worksheet similar to the following appears.



2. Save the spreadsheet as **RTD_Tutorial.xlsx** (or whichever file extension is appropriate for your version of Excel).

A note about spaces

Take note that you do not include extra spaces, as these labels will be passed directly to the XTAPI via the XTAPI RTD Server. XTAPI treats these names as attributes and does not understand that you might mean "Prod Type" and "ProdType" to be the same attribute.

Adjusting the Excel update throttling rate

Excel automatically limits the rate at which it checks for updates. By default, it sets this throttling rate to 2000 milliseconds (2 seconds). In a fast-paced trading environment, such an update rate could, and likely would, result in stale data. TT recommends that you disable the default Excel throttling rate by setting its value to 0. With this value, Excel continuously checks for updates, ensuring that you have accurate and up-to-date values.

TT provides a function called

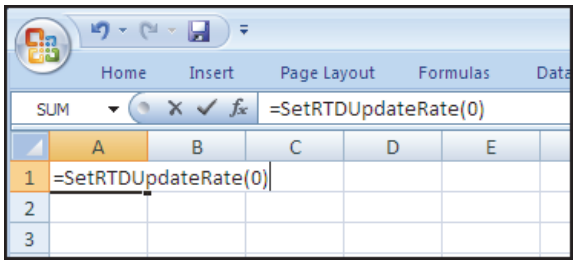
SetRTDUpdateRate()

that allows you to change the Excel throttling rate from within a spreadsheet.

Note: Changing the throttling rate with this function in any spreadsheet affects the default throttling rate for the current and all future Excel sessions.

To adust the Excel throttling rate:

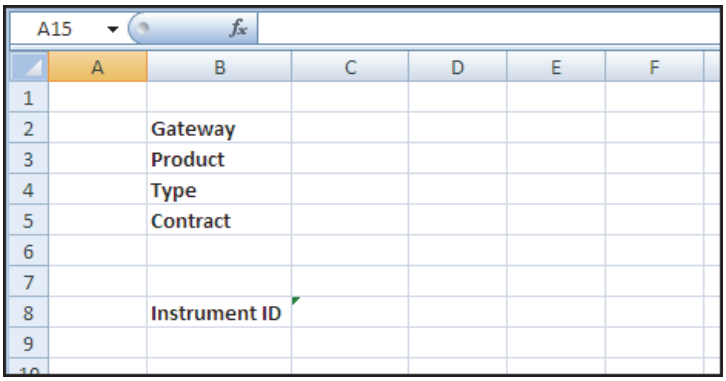
1. In your spreadhseet, place the cursor in cell A1.
2. Enter the following formula:



Specifying the contract to monitor

Organizing the layout

To start organizing the layout for the spreadsheet, you need to provide places where the trader can enter data for the query. At this point in the tutorial, you create the input section of the spreadsheet, as shown. For visual effect, leave the first column (**A**) and the first row (**1**) blank to create a frame around your spreadsheet.



Adding contract labels

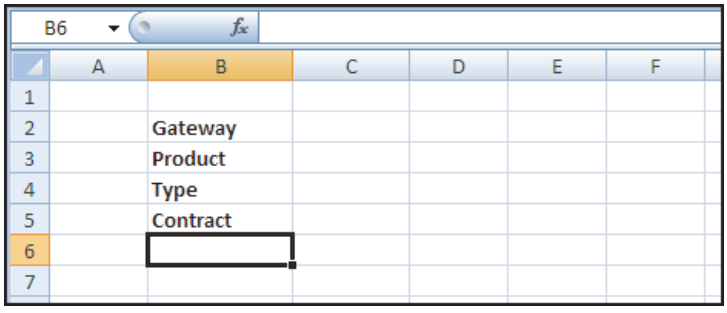
First, you need to create a place for a trader to specify contract information. Initially, the tutorial application starts by allowing a trader to enter the **Gateway**, **Product**, **Type**, and **Contract** information for an instrument. The XTAPI RTD Server uses the text in these labels to query the XTAPI for the corresponding instrument attributes, so the text you choose must match that attribute name exactly.

Later in the tutorial, you will add additional contract qualifiers.

To add labels for the contract qualifiers:

1. Place the cursor in cell **B2**.
2. Type **Gateway**; then select the text and make it bold.
3. Repeat the process for cells **B3** through **B5** with the values: **Product**, **Type**, and **Contract**, respectively.

   When finished, the spreadsheet should resemble the following.

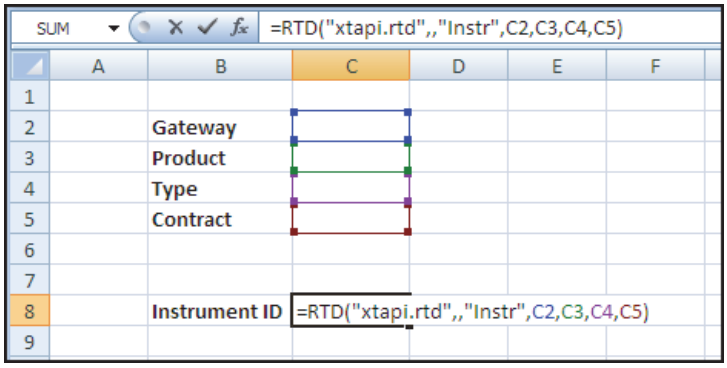


Generating an Instrument ID

Now that a trader can input contract qualifiers into the spreadsheet, you have the information to generate a unique ID that you can use to reference the instrument throughout the worksheet.

To generate an instrument ID for the specified contract qualifiers:

1. Place the cursor in cell **B8**.
2. Type **Instrument ID**; then select the text and make it bold. If necessary, resize the column to accommodate the text.
3. Place the cursor in cell **C8**.
4. Type: **=RTD("xtapi.rtd",,"Instr",C2,C3,C4,C5)**, as shown.



When you enter the formula, Excel displays **#N/A** in the cell because the contract qualifier cells do not yet contain data.

Testing the Instrument ID

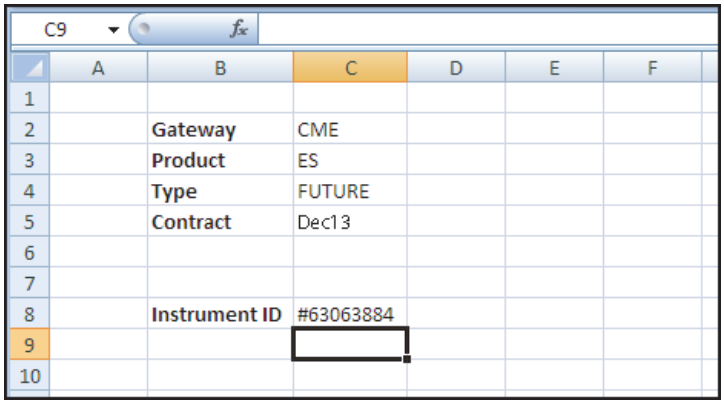
With the `RTD()` formula set to use the values from the input cells, you can now enter contract qualifiers to test your formula. After you enter values into each of the four qualifier cells, Excel updates the **Instrument ID** with an integer value.

Note: You must precede the contract name with a single quote (') to prevent Excel from treating the value as a Date data type. If you enter the date without the quote, Excel applies its default display format for dates. For example, if you enter **Dec13**, Excel displays it as **13-Dec**. When you use the **Instrument ID** in another formula, Excel would pass **13-Dec** as the contract name. Consequently, the request would fail, as **13-Dec** does not represent a valid contract name.

You can also set the Excel cell format to **Text** to stop Excel from reformatting the date.

To test the Instrument ID formula:

1. Enter the following values. If you do not use the TT CME Gateway, choose alternate contract credentials.
   - Gateway: **CME**
   - Product: **ES**
   - Type: **FUTURE**
   - Contract: **'Dec13**
2. Observe that the **Instrument ID** field in cell **C8** now contains an integer similar to the following.



	A	B	C	D	E	F
1						
2		**Gateway**	CME			
3		**Product**	ES			
4		**Type**	FUTURE			
5		**Contract**	Dec13			
6						
7						
8		**Instrument ID**	#63063884			
9						
10						

Note: As a reminder, this value has no direct connection to the actual instrument, so you cannot rely on the value remaining the same each time you open the spreadsheet.

Selecting market data to retrieve

Accessing market data

Now that you can specify a contract, you probably want to get some information about the contract as it currently trades in the market. To demonstrate how you can use the XTAPI RTD Server to get live market data, you now create a market data section in the spreadsheet.

This section uses the `RTD()` formula to query and display live market data values for the:

- Last traded price (**Last**)
- Opening price for the current trading session (**Open**)
- Highest price for the current trading session (**High**)
- Lowest price for the current trading session (**Low**)
- Closing price for the previous trading session (**Close**)

Later in the tutorial, you complete this market data section of the application.

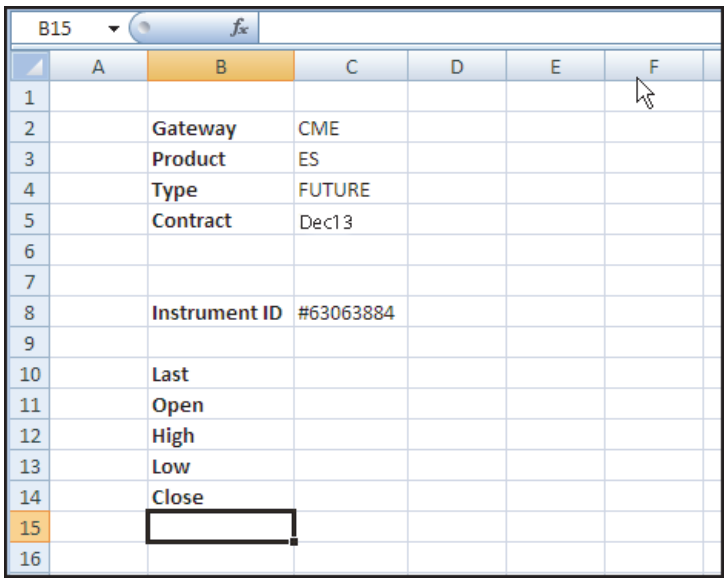
Adding market data labels

First, you create the labels for the market data. Because the formula that retrieves the data uses the label text, you must enter the labels exactly as shown so that they match the corresponding instrument attributes.

To add the labels for the market data:

1. Place the cursor in cell
2. B10
3. .
4. Type **Last**; then select the text and make it bold.
5. Repeat the process for cells **B11** through **B14** with the values **Open**, **High**, **Low**, and **Close**, respectively.

   When finished the spreadsheet should resemble the following.



Adding market data formulas

With the labels in place, you can now use the label text in the `RTD()` formula to retrieve the instrument data from the exchange, as shown in the following example.

Example: Retrieving the instrument market data using label text

```
=RTD("xtapi.rtd",,$C$8,"Last")
```

This formula instructs the XTAPI RTD Server to get the instrument identified by the ID in cell **C8** and request the instrument attribute named **Last**. While this approach works, it also limits the flexibility of the spreadsheet. If you later want to display a different value, you would have to update the formula.

By using cell references like **B10** and **B14** instead of actual names, you build flexibility into your RTD spreadsheet. The following example shows how you can use the contents of another cell in a formula.

Example: Retrieving the instrument market data using label text cell reference

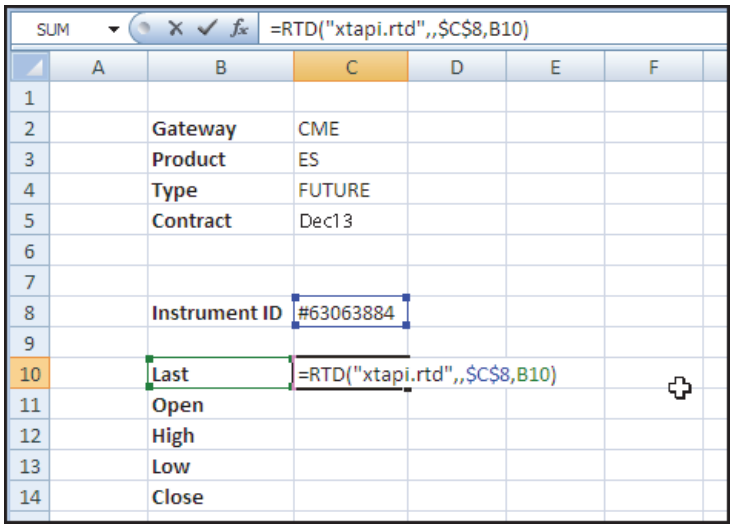
```
=RTD("xtapi.rtd",,$C$8,B10)
```

Excel interprets this formula as, "Using the instrument defined in cell **C8**, retrieve the value of the attribute specified in **B10**." Referencing the values in these cells, you instruct the XTAPI RTD Server to, "Show me the last traded price for the ES Dec13 Futures contract on CME." If you wanted to see some value other than **Last**, you need only replace the text in **B10** with the name of another XTAPI instrument attribute, such as **Open** or **High**.

The **$** notation (**$C$8**) instructs Excel to use the absolute cell reference instead of a relative one, which is the default Excel behavior. If you copy or move a formula that includes an absolute cell reference, Excel does not adjust the cell reference.

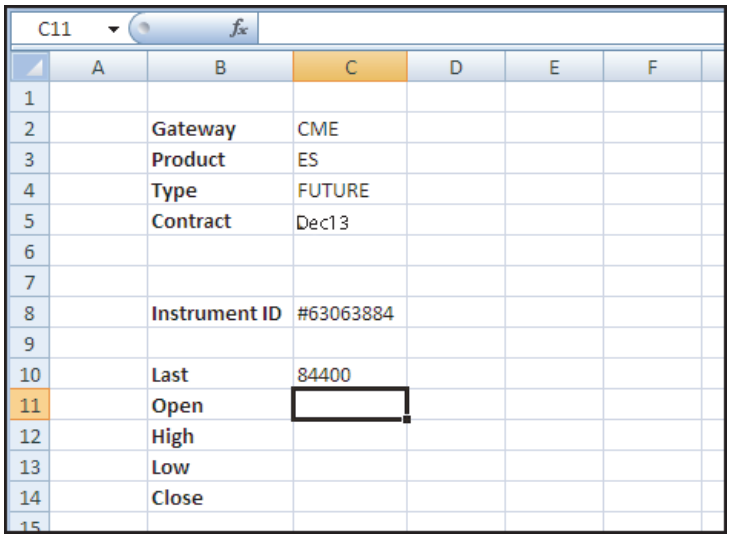
To add the formulas of the market data:

1. Place the cursor in cell **C10**.
2. Enter the `RTD()` formula as follows:



3. Press **Enter**.

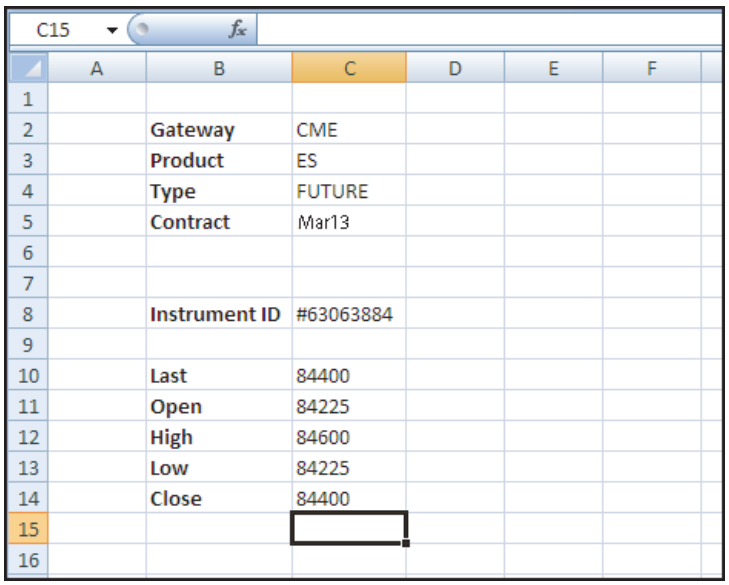
The cell contents automatically update with the last traded price for the instrument.



	A	B	C	D	E	F
1						
2		Gateway	CME			
3		Product	ES			
4		Type	FUTURE			
5		Contract	Dec13			
6						
7						
8		Instrument ID	#63063884			
9						
10		Last	84400			
11		Open				
12		High				
13		Low				
14		Close				
15						

4. In cells **C11** through **C14**, enter the same formula, changing the **B10** cell reference as appropriate. As a shortcut, you can use Excel's copy/paste or fill down functionality to supply the cell formulas.

   When finished, your spreadsheet looks similar to the following. Also, based on market activity, you should see the values change to reflect real-time changes in the market.



	A	B	C	D	E	F
1						
2		Gateway	CME			
3		Product	ES			
4		Type	FUTURE			
5		Contract	Mar13			
6						
7						
8		Instrument ID	#63063884			
9						
10		Last	84400			
11		Open	84225			
12		High	84600			
13		Low	84225			
14		Close	84400			
15						
16						

Testing your application

Trying different contracts

Now that your spreadsheet can display live price data for contracts, you can change the contract qualifiers to see how your RTD application automatically adjusts its display to reflect changing contract identifiers. When you change one or more of the values in cells **C2**-**C5**, the **Instrument ID** and the market data values update as you change each cell.

To see how the spreadsheet updates as you change values, try changing values as follows. If you connect to a different TT Gateway, adjust the values accordingly.

Change...	To...	Result
Contract	Mar13	(Changing to a new valid contract) **Instrument ID** updates with a new unique identifier Price values update to reflect the new contract
Product	EJ	(Changing to a new product that trades the same contract name) **Instrument ID** updates with a new unique identifier Price values update to reflect the new contract
Product	G0	(Changing to a product that does not trade the same contract name) **Instrument ID** updates with a new unique identifier Price values change to **#N/A** because the EJ product does not trade a Mar10 contract
Contract	Jul09	(Changing to a valid contract for the product) **Instrument ID** updates with a new unique identifier Price values update to reflect the new contract

If you can access additional TT Gateways through X_TRADER®, you can also change the **Gateway** value and try other contracts.

Tip: If you want to verify the accuracy of the data, you can open an X_TRADER® Market Grid and compare the values.

Saving your work

Before continuing to the next chapter, save your work.

4 Displaying Market Depth

Chapter overview

This chapter focuses on retrieving live market depth for an instrument. It also explains how to use multi-dimensional arrays in Excel to simplify the data retrieval process.

In this chapter

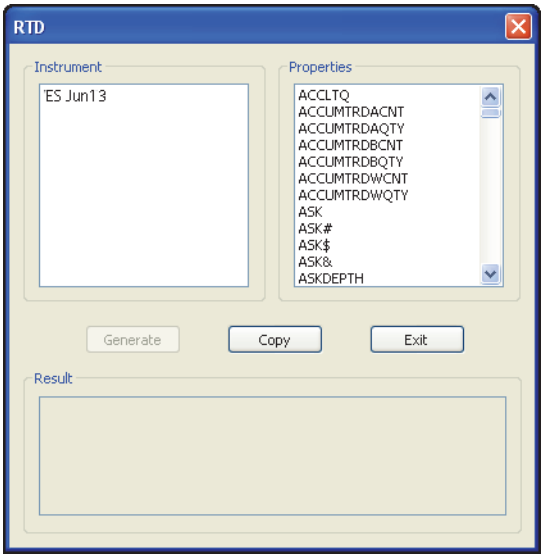
Overview

In review

In the previous chapter, you learned how to extract single values (Open, High, etc.) from the XTAPI RTD engine and to display them in the spreadsheet. Because you linked these attributes to a single instrument definition, Excel updates these values in real time as they change in a live market. You can now easily monitor these and other live values for attributes of any tradable contract.

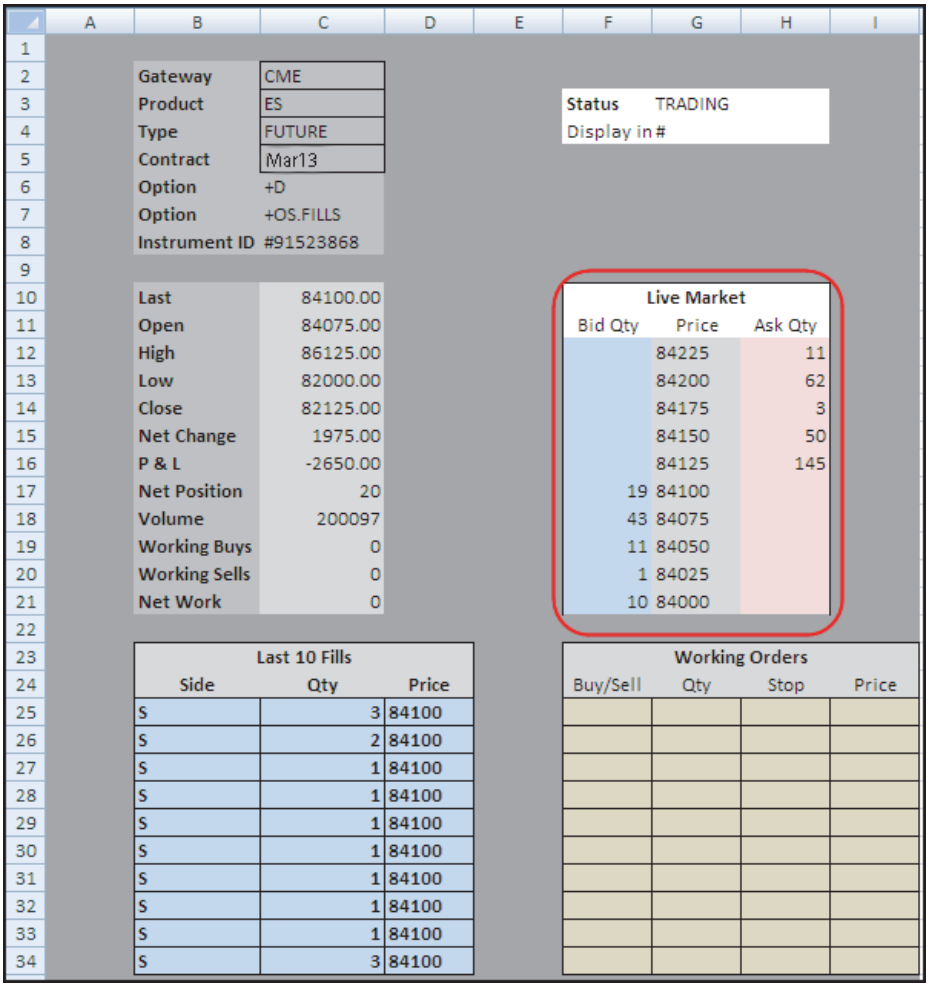
For a full list of instrument attributes you can monitor through the XTAPI RTD Server, refer to the *XTAPI RTD Server Feature Guide*.

If you are also running the X_TRADER® application, you can use the RTD Generator window to display the available attributes for a contract, as shown. For information about the RTD Generator, refer to the X_TRADER® Help System.



Mimicking the MD TRADER® display

The XTAPI RTD Server can display more than just single values. Using more advanced Excel functionality, you can display market data in a similar way to TT's MD TRADER®. This tutorial shows you how to create and display a depth display flanked by live updates for Bid and Ask quantities, as follows:



Gateway	CME
Product	ES
Type	FUTURE
Contract	Mar13
Option	+D
Option	+OS.FILLS
Instrument ID	#91523868

Status TRADING
Display in #

Last	84100.00
Open	84075.00
High	86125.00
Low	82000.00
Close	82125.00
Net Change	1975.00
P & L	-2650.00
Net Position	20
Volume	200097
Working Buys	0
Working Sells	0
Net Work	0

Live Market

Bid Qty	Price	Ask Qty
	84225	11
	84200	62
	84175	3
	84150	50
	84125	145
19	84100	
43	84075	
11	84050	
1	84025	
10	84000	

Last 10 Fills

Side	Qty	Price
S	3	84100
S	2	84100
S	1	84100
S	1	84100
S	1	84100
S	1	84100
S	1	84100
S	1	84100
S	1	84100
S	3	84100

Working Orders

Buy/Sell	Qty	Stop	Price

Leveraging Excel for multi-dimensional arrays

How Excel processes single values

Before starting to build the MD TRADER® display, you should understand some basic theory relating to how Excel processes XTAPI RTD Server data.

When you extract a single value, such as the **Open** attribute from the last chapter, the Excel spreadsheet sends a request to the XTAPI RTD Server asking, "For the specified instrument, what is the value of its **Open** attribute?" The XTAPI RTD Server then forwards the request to an instance of the XTAPI, which processes the request and places the value in a memory space managed by RTD. Excel continuously monitors this memory space and displays the value in the appropriate cell.

When the TT Gateway publishes a new value for **Open**, the XTAPI places the new value into the previously designated memory space and informs Excel that it needs to refresh its display. Conceptually, keeping live market data up-to-date involves the following tasks:

- XTAPI connects to a TT Gateway and processes market data updates.
- XTAPI RTD Server queries the XTAPI for live market data.
- Excel keeps refreshing the spreadsheet display.

Using in-memory arrays for multiple values

When it comes to a more complex request like market depth, you actually ask RTD and XTAPI to give you not just one value, but a large set of values—the entire depth of book. For each level of depth you request, Excel receives the following values:

- Quantity of Bids
- Quantity of Asks
- Price level associated with each Bid and Ask

Of course, Excel cannot store all of this information in a single memory location as it does the **Open** value. Therefore, you need a way to reserve not just a single memory space for the data, but an entire array of memory spaces.

Because the information you want for market depth is naturally organized into rows and columns (bids in one column, asks in another, and prices in a third), placing the data into a spreadsheet would be ideal. At first glance, you might think you should just put a formula in each of the cells you want to fill with retrieved data. For example, you could put a formula to return the first level of depth in a cell, then copy that formula into each of the other cells and change the attribute as desired.

However, such an approach would be inefficient. Excel can refresh the screen data at a rate of several hundred times per second and would need to re-evaluate every cell that RTD references. In other words, Excel would need to call the `RTD()` function in every cell to retrieve market data, and would call all of the functions over and over again each second, which would cause an undue burden on the system.

Using more advanced Excel functionality for temporary in-memory multi-dimensional arrays provides a more robust and efficient means to process the market data.

Understanding the RTD formula for requesting market depth

The `RTD()` function call you use to request market depth looks much like the one you use to request single values, except that in addition to the instrument attribute, you specify a place in memory to store the data. The following example shows how to request five levels of market depth and store it in an in-memory array:

```
=RTD("xtapi.rtd",,$C$8,"PriceDepth(5)",CELL("Address",A1))
```

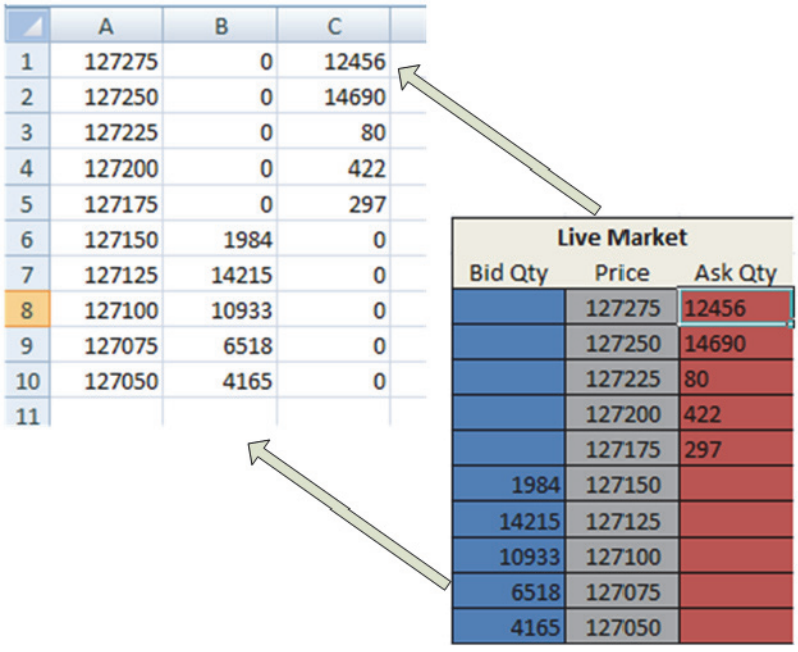
By now, you should be able to interpret most of this formula. It asks the XTAPI RTD Server to return the data for the **PriceDepth** attribute of the instrument defined in cell **C8**. The **5** argument for **PriceDepth(5)** call indicates that you want five levels of Ask depth and five levels of Bid depth.

Because the requested data contains more than a single value, Excel stores the results in a temporary, multi-dimensional memory space. You can think of it as a hidden spreadsheet that is private for this data request. When you request the market depth in this manner, XTAPI assembles all of the depth information and returns it to the XTAPI RTD Server as a single block of data. Because the data cannot fit into a single cell memory space, the XTAPI RTD Server creates the hidden spreadsheet and places the data into it, with each piece of data in its own spreadsheet cell. Instead of making one `RTD()` function call to the XTAPI for each cell in the matrix, this approach greatly reduces the communication between Excel and the TT Gateway, which greatly improves performance.

You can now reference the returned data within this hidden spreadsheet directly from your cells by specifying the **CELL("Address", A1)** parameter to your formula. In this example, you ask Excel to place the value of cell **A1** from the hidden, temporary spreadsheet into this cell in your spreadsheet. If you were to replace **A1** with **B3**, the value of cell **B3** would display in the cell.

Retrieving the data from the array

You determine the amount of data to request from the XTAPI by the number you specify for the **PriceDepth** attribute. In this example, you specify **PriceDepth(5)**; then the XTAPI RTD Server returns five levels of depth on each side of the market, where column **A** contains the prices, column **B** contains the Bid quantities, and column **C** contains the Ask quantities.



	A	B	C
1	127275	0	12456
2	127250	0	14690
3	127225	0	80
4	127200	0	422
5	127175	0	297
6	127150	1984	0
7	127125	14215	0
8	127100	10933	0
9	127075	6518	0
10	127050	4165	0
11			

Live Market		
Bid Qty	Price	Ask Qty
	127275	12456
	127250	14690
	127225	80
	127200	422
	127175	297
1984	127150	
14215	127125	
10933	127100	
6518	127075	
4165	127050	

By specifying the **CELL("Address",A1)** parameter in the formula, you instruct Excel to use cell **A1** of the hidden spreadsheet as the starting location for the data. When it receives the data, Excel populates the hidden cells just like pasting an array of values into a regular spreadsheet, where each value occupies its own cell. You can think of the returned data as a three-column, ten-row spreadsheet and can reference the different values just like you would in a regular spreadsheet. For example, if you specify **CELL("Address",C2)**, Excel returns the value in the third column of the second row (in this case, the number of Asks four ticks away from the inside market, or 14690).

Transforming the columns

As you might have noticed, MD TRADER® orders the data a little differently than the data returned by the **PriceDepth** attribute. Because this tutorial tries to mimic MD TRADER®, you need to change which columns the spreadsheet uses to display the data, which is explained in the section called Creating the Live Market data grid, on page 29. The XTAPI RTD Server returns all of the data in a single call, so switching columns A and B is simple; you need only put the data where you want it. So you can arrange the data in the order **Bid Qty**, **Price**, **Ask Qty** or in the order **Price**, **Bid Qty**, **Ask Qty** simply by changning the order in which you reference the columns.

Setting up the live market data section

About depth updates

As an optimization, the XTAPI RTD Server does not request depth updates when accessing instrument data unless you specifically want it. In most markets, many updates to the Bid and Ask quantities occur away from the inside market; so unless you want to access market depth, you need not burden the system with excessive update notifications. While enabling depth updates does increase the traffic and load on the system, it does not noticeably affect system performance. If you want to create an application that accesses only a few instruments strictly for gathering information, using the XTAPI RTD Server instead of a full-blown MD TRADER® window conserves system resources. However, unless you need depth, you have no reason to include it.

The XTAPI RTD Server allows you to pass options in the `RTD()` function call. The **D** option controls whether the XTAPI RTD Server requests and processes depth updates, as follows:

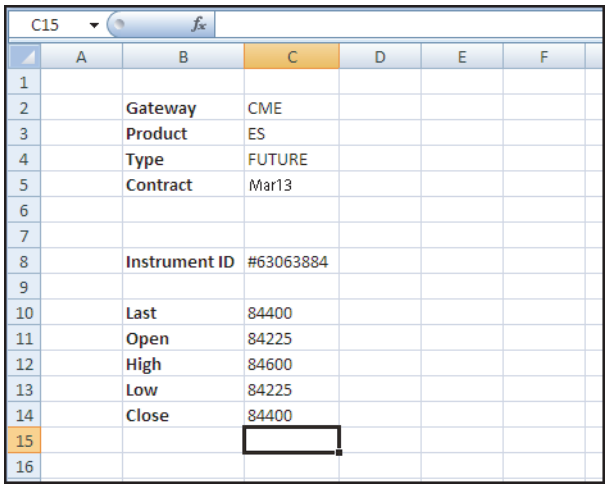
- **+D** enables depth updates
- **-D** disables depth updates (default)

Adding a depth option to the spreadsheet

To enable depth updates, you need to add the option to the formula that requests an instrument. In this tutorial, you need to modify the `RTD()` call in cell **C8** that generates the Instrument ID. To do so, you fill in cells **B6** and **C6** that you left blank in the last chapter.

To add a depth option to the spreadsheet:

1. Open the Excel spreadsheet.

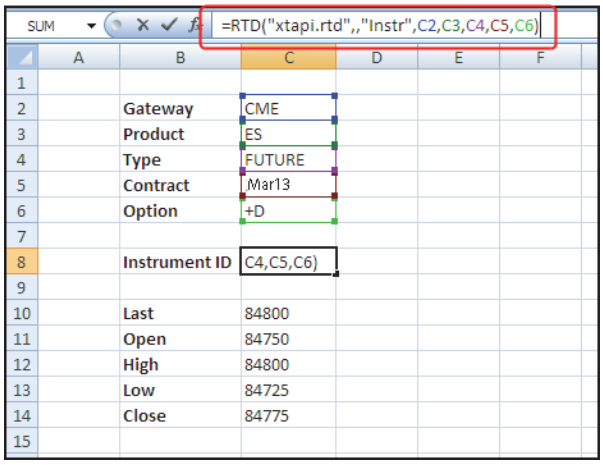


Note: Because Excel automatically calculates formulas when it starts, you should notice that the value for the Instrument ID differs from when you last opened the spreadsheet. However, the new value works correctly for the this session and also illustrates why you should never use the generated number directly in your formulas.

2. Place the cursor in cell **B6**, and type **Option**.
3. Place the cursor in cell **C6**, and type **'+D**.

Note: As a reminder, you must include the single quote (') to prevent Excel from interpreting the **+** as an arithmetic operator.

4. Place the cursor in cell **C8**.
5. Update the formula to add the value of cell **C6**, as follows:



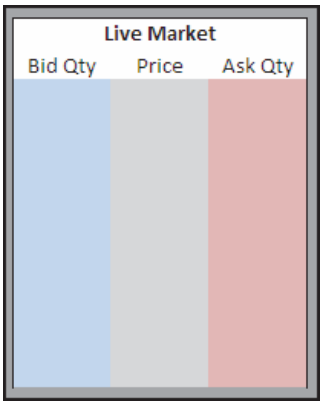
The XTAPI RTD Server can now receive market depth updates for the instrument.

6. Save, but do not close, the spreadsheet.

Creating the Live Market data grid

Overview

Now that you enabled the XTAPI RTD Server to receive market depth updates for the instrument, you need a place to display the values.



Defining and labeling the market grid

To help visualize the market, you can label the grid area and assign different background colors for the columns.

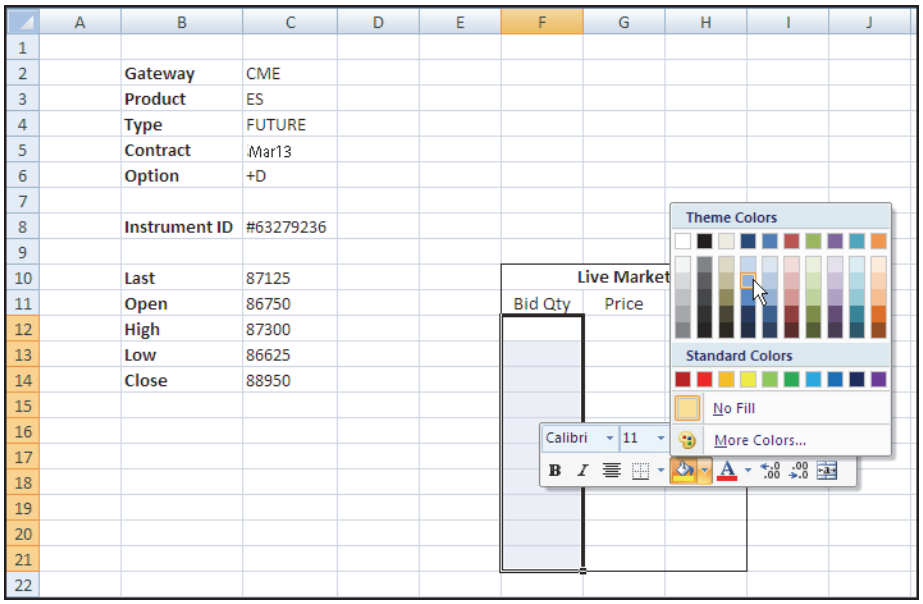
To define and label the market grid:

1. Select cells **F10**-**H21**; then from the **Borders** menu, choose **Outside Borders**.
2. Place the cursor in cell **G10**, and enter **Live Market** as bold text.
3. To center the grid title, select cells **F10**-**H10**; then from the **Alignment** toolbar, select **Merge & Center**.

   Excel centers the label across the three columns.
4. Enter and center the column labels **Bid Qty**, **Price**, and **Ask Qty** in cells **F11**-**H11**, respectively.

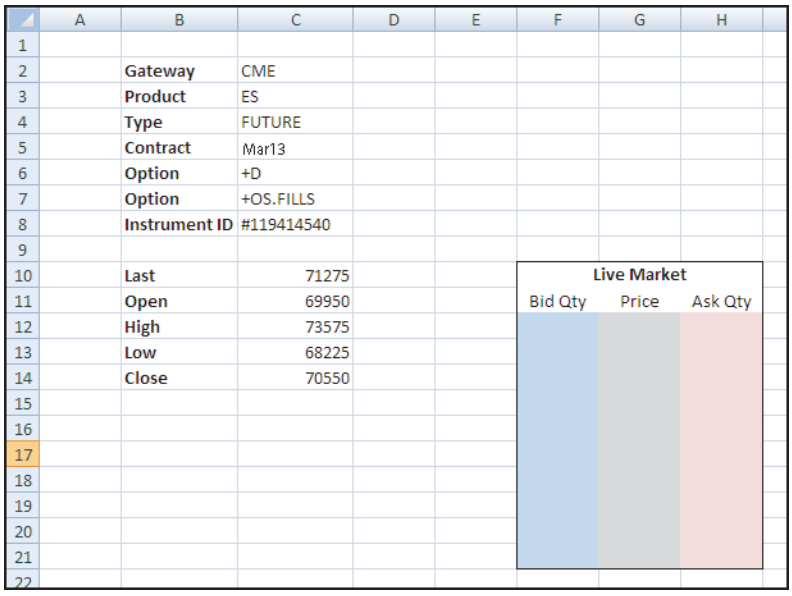
	A	B	C	D	E	F	G	H
1								
2		Gateway	CME					
3		Product	ES					
4		Type	FUTURE					
5		Contract	Mar13					
6		Option	+D					
7								
8		Instrument ID	#63279236					
9								
10		Last	87125			Live Market		
11		Open	86750			Bid Qty	Price	Ask Qty
12		High	87300					
13		Low	86625					
14		Close	88950					
15								
16								
17								
18								
19								
20								
21								
22								

5. To change the color for the **Bid Qty** column, select cells **F12**-**F21**; then change the background color for the selected cells as desired. For example, you can use the context menu to access the **Fill** color, as shown:



6. Repeat the process to change the background colors for the **Price** and **Ask Qty** columns of the market grid.

   When finished, your spreadsheet should look similar to the following:



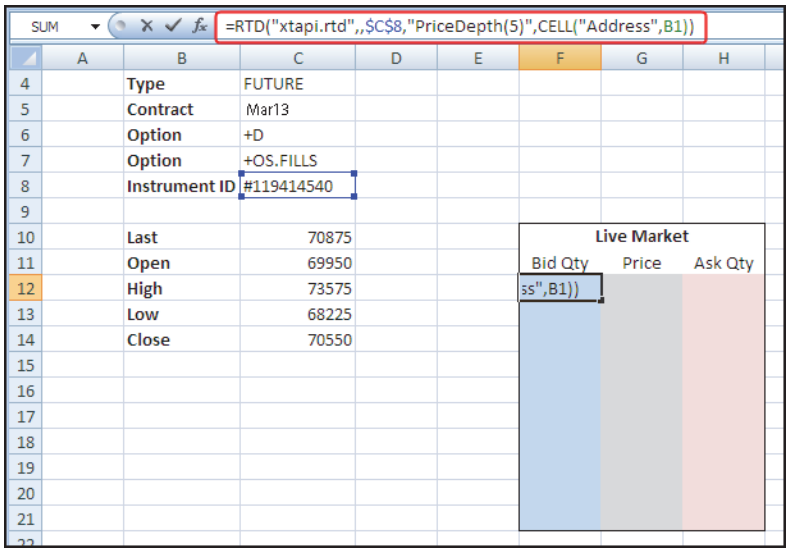
Adding the market data formulas to the grid

Now that you have the market grid in place, you are ready to enter the RTD formulas to display the live market data.

▶ To add the market data formulas to the grid:

1. Starting with the **Bid Qty**, place the cursor in cell **F12**.
2. Enter the following formula:

```
=RTD("xtapi.rtd",,$C$8,"PriceDepth(5)",CELL("Address",B1))
```



As a reminder, this formula:

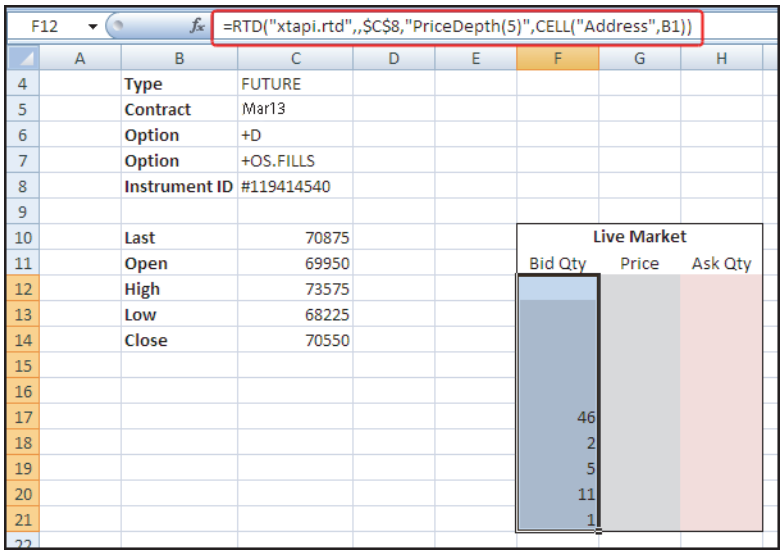
- Requests five levels of market depth for the instrument specified in cell **C8**.
- Places the values in a temporary in-memory spreadsheet.
- Adds the contents of the in-memory worksheet's cell **B1** into cell **F12** of the main worksheet. You use **B1** in the formula because the in-memory **PriceDepth** array puts the **BidQty** values in column **B**.

3. Observe that the formula appears to have no effect.

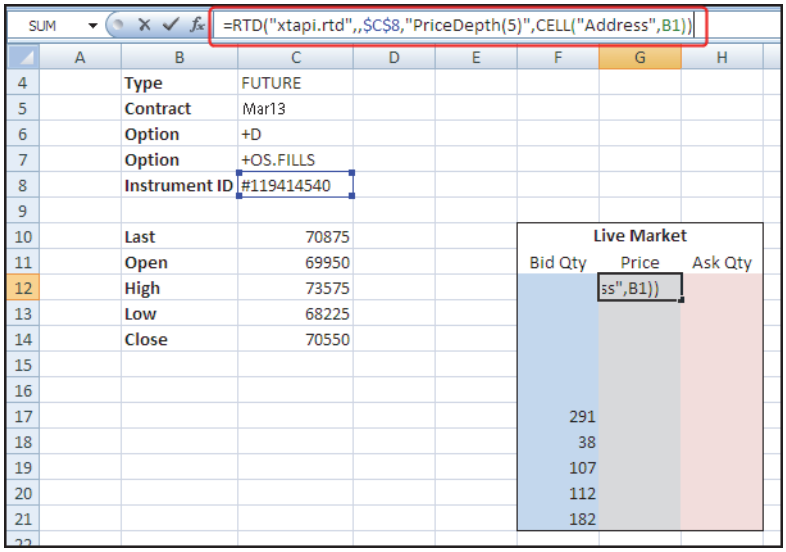
   The **Live Market** grid centers the inside market values; therefore **Bid Qty** displays no values above the inside market, only *at* or *below* it.

4. To display the bid depth quantities, place the cursor in cell **F12** and select cells down through cell **F21**; then type `Ctrl+D` to fill-down (copy the formula to each of the selected cells).

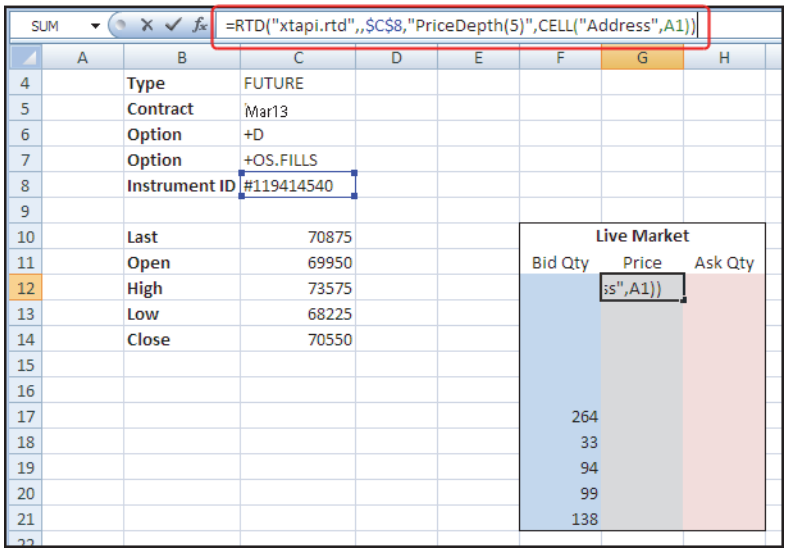
   The bottom five cells now show live bid quantities, similar to the following:



5. To create the depth display, you can reuse the formula in cell **F12** and update it to use the values in column **A** instead of column **B**. Select cell **F12**; then copy and paste the formula from cell **F12** to cell **G12**. (Do not use the Excel "fill right" functionality, as it also copies the cell background color.)

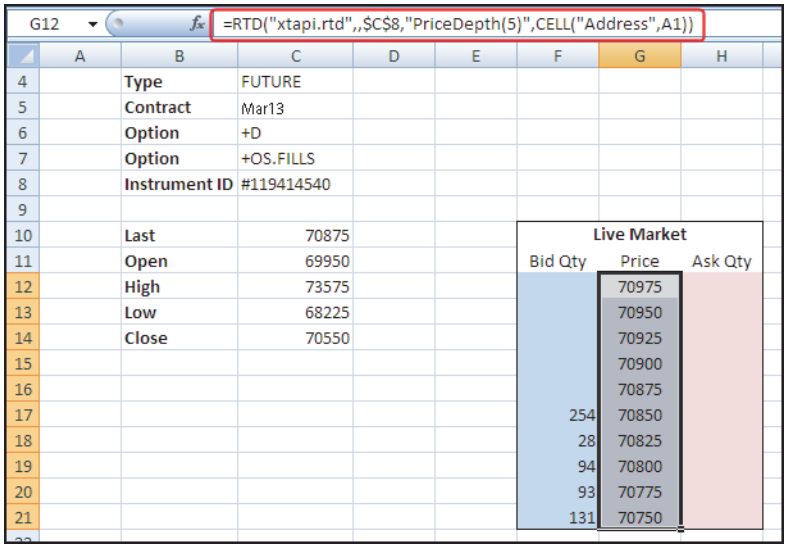


6. Select cell **G12**, and change the formula to reference column **A1** of the in-memory spreadsheet, which contains the prices, as shown:



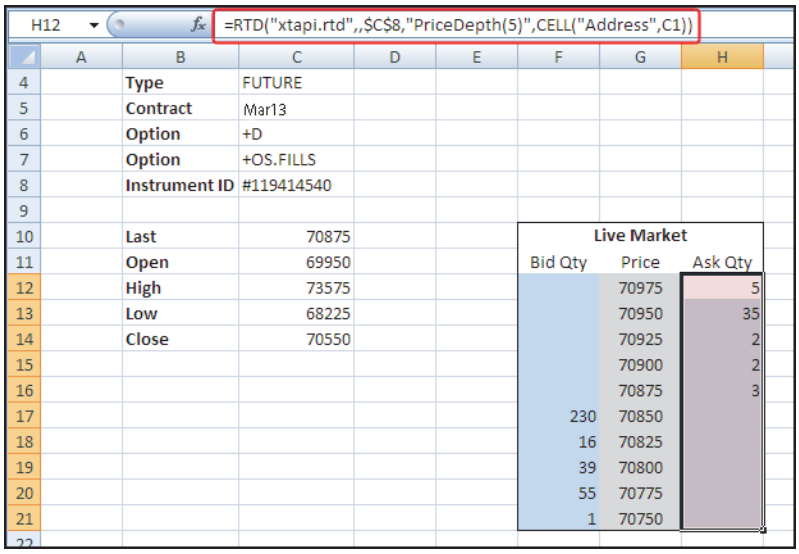
7. Place the cursor in cell **G12**; then select cells **G12**-**G21**. As before, type `Ctrl+D` (or use standard copy/paste) to copy the formula down to the selected cells.

   The prices appear in the column cells, similar to the following:



8. To finish the grid and include the Ask quantities, copy the formula from cell **G12** to cell **H12**. Then repeat the process of changing the column reference to **C1** and copying the formulas to cells **H13**-**H21**.

   When finished, the **Live Market** grid resembles the following.



Testing your application

Exploring market conditions

To test your application:

1. Enter various contract qualifiers to see the market data update.
2. In X_TRADER®, submit Buy and Sell orders if the instrument has no orders working in the market.
3. In your RTD application, observe the changes in the **Live Market** grid.

Saving your work

Before continuing to the next chapter, save your work and exit the Excel application.

5 Displaying Fills and Working Orders

Chapter overview

This chapter expands the spreadsheet functionality by showing you how to use order sets to retrieve fill and working order information for an instrument.

In this chapter

Overview

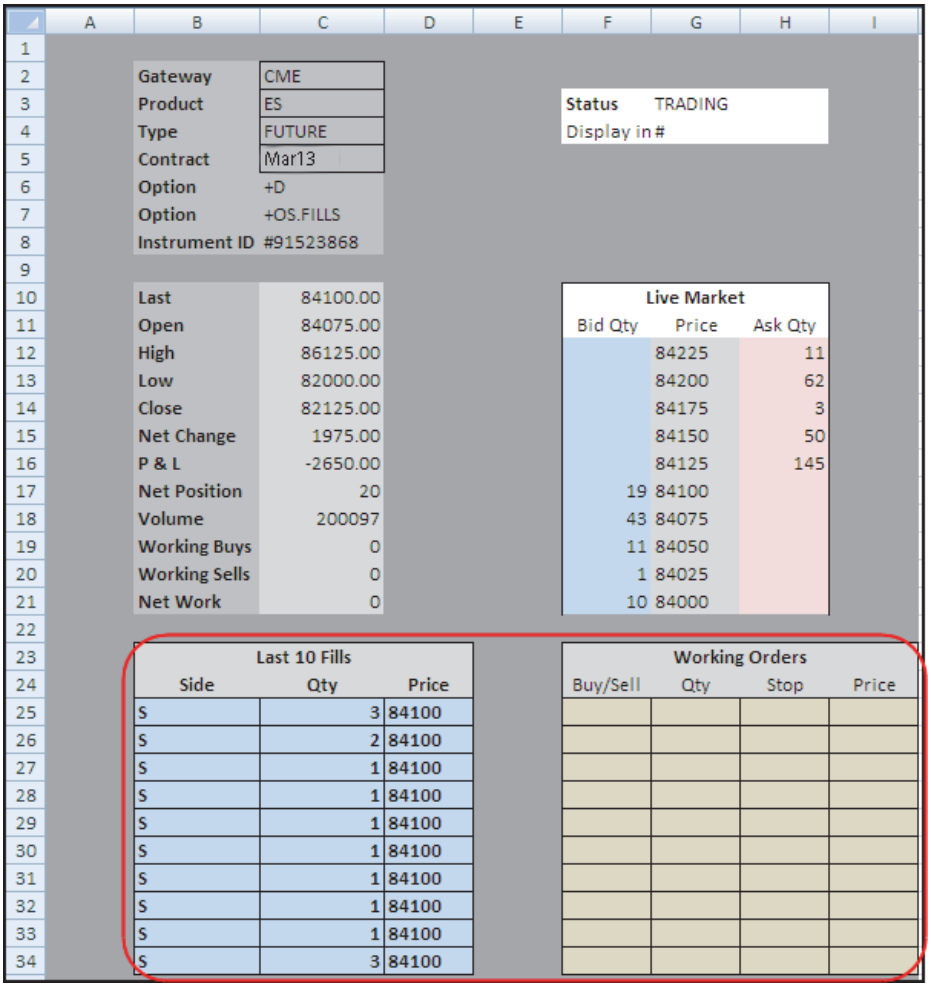
In review

In the previous chapter, you learned to use the hidden spreadsheet feature in Excel to extract the value of a single instrument attribute. You can also use hidden spreadsheets (as described in the section called, Retrieving the data from the array) to store values for multiple attributes retrieved from the XTAPI RTD Server.

For a full list of instrument attributes you can monitor through the XTAPI RTD Server, refer to the *XTAPI RTD Server Feature Guide*.

Displaying fills

In this chapter, you use the a hidden spreadsheet again when requesting fill records and a list of working orders, as shown.



Gateway	CME
Product	ES
Type	FUTURE
Contract	Mar13
Option	+D
Option	+OS.FILLS
Instrument ID	#91523868

Status TRADING
Display in #

Last	84100.00
Open	84075.00
High	86125.00
Low	82000.00
Close	82125.00
Net Change	1975.00
P & L	-2650.00
Net Position	20
Volume	200097
Working Buys	0
Working Sells	0
Net Work	0

Live Market

Bid Qty	Price	Ask Qty
	84225	11
	84200	62
	84175	3
	84150	50
	84125	145
19	84100	
43	84075	
11	84050	
1	84025	
10	84000	

Last 10 Fills

Side	Qty	Price
S	3	84100
S	2	84100
S	1	84100
S	1	84100
S	1	84100
S	1	84100
S	1	84100
S	1	84100
S	1	84100
S	1	84100
S	3	84100

Working Orders

Buy/Sell	Qty	Stop	Price

As you did in retrieving price updates in the last chapter, you can ask the XTAPI RTD Server to store the fill information in another temporary, hidden spreadsheet. You also use cell references to access the fill data, just like you did with the price depth information. When changes in market data cause Excel to refresh its display, the XTAPI RTD Server updates the appropriate cells in the main spreadsheet with the values from the temporary spreadsheet.

When you requested price depth, the XTAPI RTD Server sent data representing the price levels, bid quantities, and ask quantities. You used a single variable, but received three distinct sets of data. When requesting fill information, however, you can combine the request for multiple attributes values into a single `RTD()` formula call and store the results in a new temporary, in-memory spreadsheet.

For this application, you ask the XTAPI RTD Server for the following fill information:

- Side of the market (Buy or Sell)
- Fill quantity
- Fill price

Working with order sets

About order sets

An XTAPI RTD Server application accesses fill data through an XTAPI order set. By default, XTAPI order sets do not retain fill records after processing, which means that the fill information you need for this application is not available unless you request otherwise. For your application to access fill information, you must instruct the XTAPI order set to retain its fill records. You enable this option in a manner similar to the way you enabled price depth in the last chapter.

To extract price depth, you added the **+D** option to the Instrument ID generation formula that told the XTAPI RTD Server to enable market depth. Similarly, you can add the **+OS.FILLS** option to the formula to enable fill retention.

More about the +OS option

The **+OS** part of this option requires some explanation. XTAPI allows users to create multiple order sets. Order sets are logical groupings of orders that you can use to isolate different strategies or to determine values like P&L in a more granular fashion. When the XTAPI RTD Server makes a connection to XTAPI, it automatically creates a new order set. The XTAPI manages much of its data through order sets, such as P&L values, whether or not to retain Fill receipts, whether to permit order routing, and so on. Because fill receipts are turned on or off at the order set level, the RTD option for enabling fill retention is a sub-option of the order set. The **+OS.FILLS** option provides RTD a shorthand way of enabling fills in an order set. For a full list of order set options, refer to the *XTAPI RTD Server Feature Guide*.

Enhancing the RTD formula

To retain fills in the order set, you need only to add the option to the existing instrument definition. You can use the same technique you used to add the depth option; namely, adding the option to cell **C7** and referencing the cell in the formula, as follows.

```
=RTD("xtapi.rtd",,"Instr",C2,C3,C4,C5,C6,C7)
```

Adding a fills option to the spreadsheet

To retrieve fill information, you need to add the option requesting the fills from the order set. In this tutorial, you need to modify the `RTD()` formula in cell **C8** that generates the Instrument ID. To do so, you use cells **B7** and **C7** that you left blank in the last chapter.

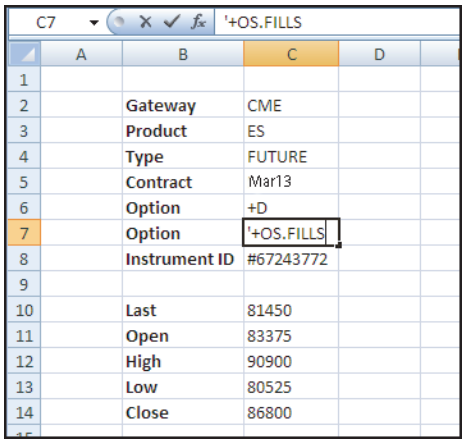
▶ To add a fills option to the spreadsheet:

1. Open the Excel spreadsheet, if necessary.

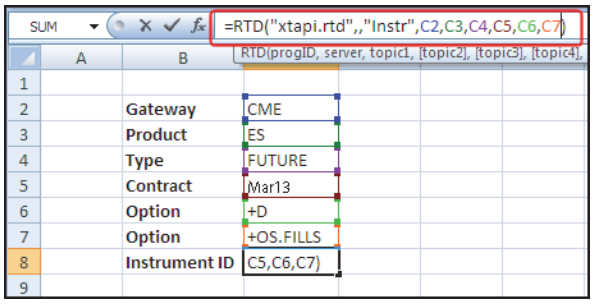
	A	B	C	D	E
1					
2		Gateway	CME		
3		Product	ES		
4		Type	FUTURE		
5		Contract	Mar13		
6		Option	+D		
7					
8		Instrument ID	#67243772		
9					
10		Last	81450		
11		Open	83375		
12		High	90900		
13		Low	80525		
14		Close	86800		

2. Place the cursor in cell **B7**, and type **Option**.
3. Place the cursor in cell **C7**, and type **'+OS.FILLS**.

> **Note:** As a reminder, you must include the single quote (**'**) to prevent Excel from interpreting the **+** as an arithmetic operator.



4. Place the cursor in cell **C8**.
5. Update the formula to add the value of cell **C7**, as follows:



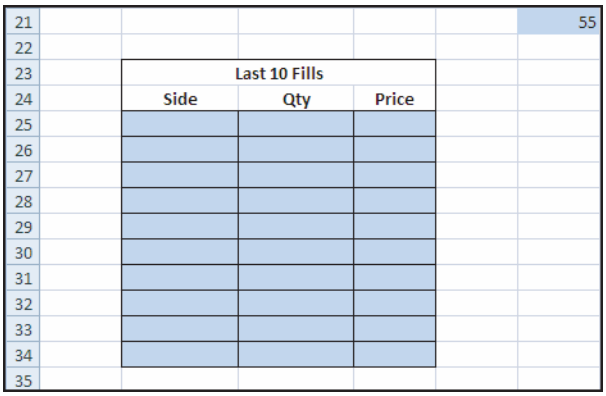
The XTAPI RTD Server now receives fill notifications for this instrument.

6. Save, but do not close, the spreadsheet.

Creating the last fills grid

Overview

Now that the instrument identified in cell **C8** provides depth information for the price display and all fills associated with the instrument, you can load the fill information in the spreadsheet. In this case, you add another grid below the existing market data values, as shown.



Note: Because you specify the instrument referenced in cell **C8**, the XTAPI RTD Server retrieves fills only for this instrument, not all the fills associated with all instruments of the MGT. Later in this tutorial, you create a global fill book for that purpose.

Limiting the number of fills

The `RTD()` formula for extracting fills follows the same pattern as the formula for extracting price data. In this case, because you want the side, quantity, and price for each of the ten most recent fills, you use the following formula:

```
=RTD("xtapi.rtd",,$C$8,"OS.FILLS(last*10)(BuySell,Qty,Price)",CELL("Address",A1)))
```

The **(last*10)** argument limits the number of fills the XTAPI RTD Server returns. With no argument, the XTAPI RTD Server returns every fill in the order book that is associated with the instrument in cell C8. Because you have limited screen space, the application displays only the ten most recent fills. As you might surmise, the XTAPI RTD Server reserves two special keywords, **last** and **first**. You can use these keywords to indicate a starting point. In this case, **last*10** always delivers the ten most recent fills, while **first*10** always delivers the first ten fills. By using **last*10**, you don't have to handle scrolling lists or to organize the fills, because your spreadsheet always displays the ten most recent fills.

Selecting fill attributes

The XTAPI tracks a lot of information about each fill it receives and stores that data in fill attributes. To specify which attributes you want to receive, you simply list them as an argument to the **OS.FILLS** option. For this application, you need only three:

- **BuySell**, which indicates whether the fill represents a Buy (B) or Sell (S) order
- **Qty**, which indicates how many contracts comprise this fill
- **Price**, which represents that price at which the contracts filled

You can access any of the other fill attributes, as well. For a list of fill attributes supported by the XTAPI, refer to the *XTAPI RTD Feature Guide*.

More about the new temporary spreadsheet

At the end of the formula, you again include that reference to a temporary worksheet. You might notice that the formula uses links to cell **A1** of the hidden worksheet just like the price depth formula. While both formulas reference the same cell, no collision occurs because each unique formula creates its own in-memory spreadsheet. Because each reference in the price display used the same formula, **PriceDepth(5)**, the XTAPI RTD Server requests the information from the XTAPI only once.

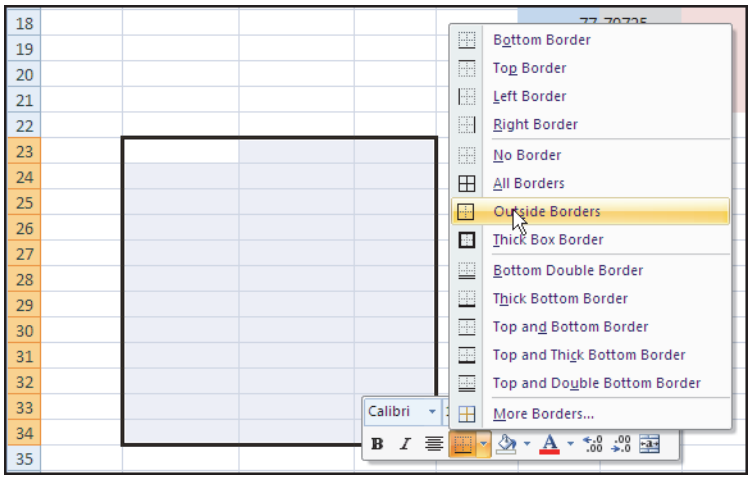
As this `RTD()` call uses a different option, **OS.FILLS**, Excel creates a new, hidden spreadsheet tied to the formula. Thus, you can again ask the XTAPI RTD Server to place its returned data in cell **A1** without worrying about overwriting other values.

Defining and labeling the fills grid

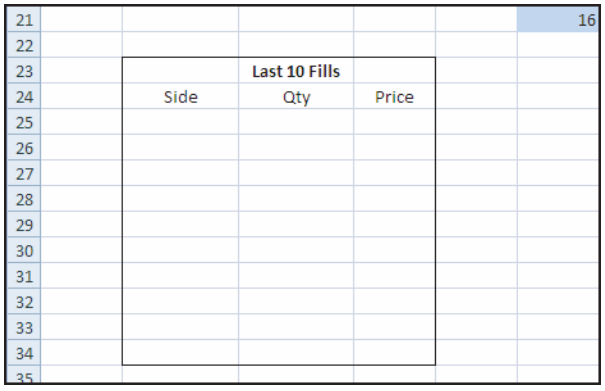
To help visualize the market, you can label the grid area and assign different background colors for the columns.

To define and label the fills grid:

1. Select cells **B23**-**D34**; then from the **Borders** menu, choose **Outside Borders**.



2. Place the cursor in cell **C23**, and enter **Last 10 Fills** as bold text.
3. To center the grid title, select cells **B23**-**D23**; then from the **Alignment** toolbar, select **Merge & Center**.
4. Enter and center the column labels **Side**, **Qty**, and **Price** in cells **B24**-**D24**, respectively.

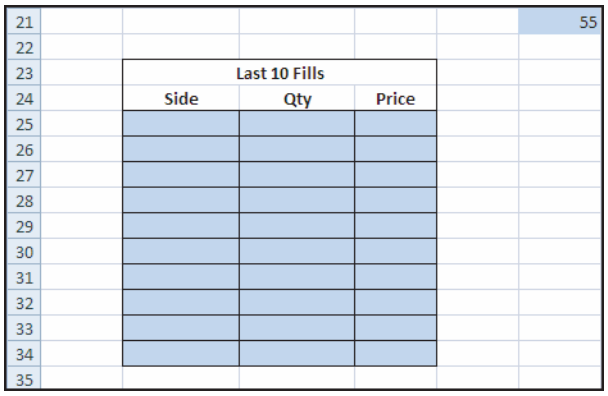


5. To change the appearance of this grid, select cells **B25**-**D34**; then do the following:

   1 To change the border style, from the **Borders** menu, choose **All Borders**.

   2 To change the background color, choose a different **Fill** color.

   When finished, your spreadsheet should look similar to the following:



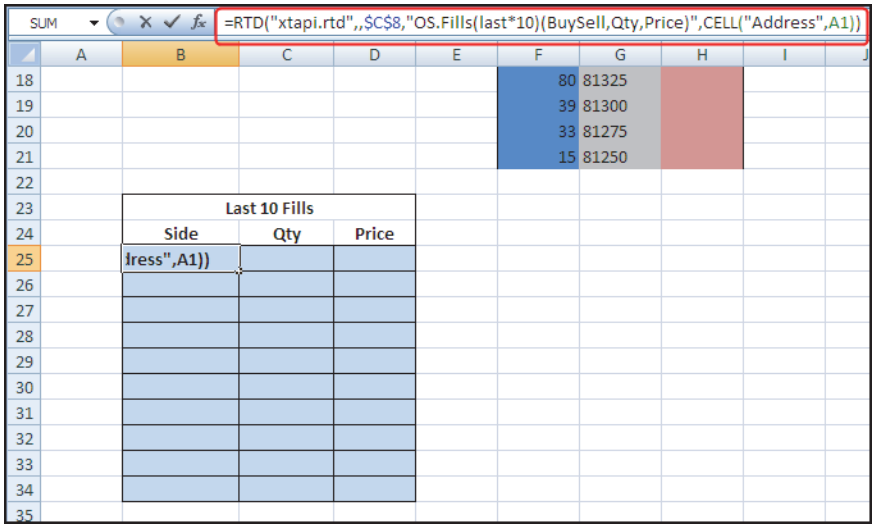
Adding the fill data formulas to the grid

Now that you have the **Last 10 Fills** grid in place, you can enter the RTD formulas to display the live market data.

To add the fill data formulas to the grid:

1. Starting with the **Side** column, place the cursor in cell **B25**.
2. Enter the following formula:

```
=RTD("xtapi.rtd",,$C$8,"OS.Fills(last*10)(BuySell,Qty,Price)",CELL("Address",A1))
```



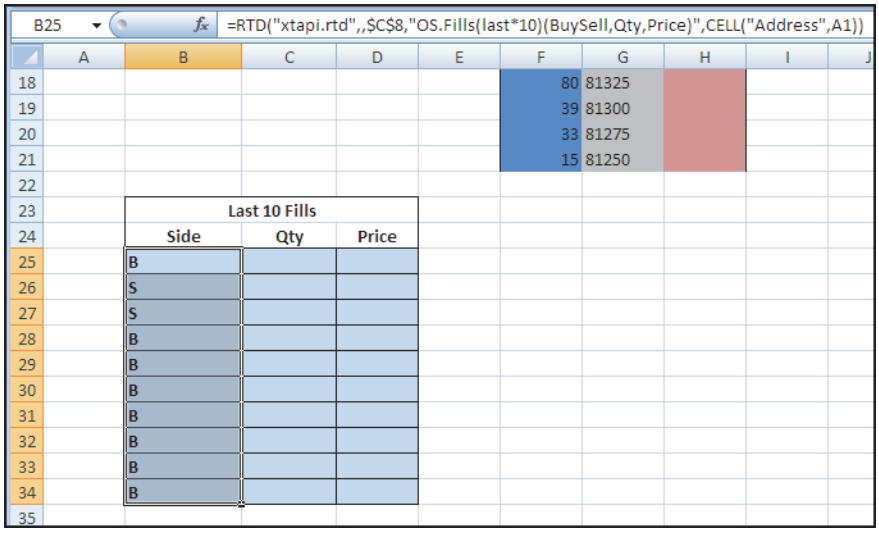
Note: If you happen to have filled working orders in the market, the cell populates with either **B** or **S**, as appropriate.

As a reminder, this formula:

- Requests the **BuySell**, **Qty**, and **Price** attributes of the last ten fills for the instrument specified in cell **C8**
- Places the values in a temporary in-memory spreadsheet
- Adds the contents of the in-memory worksheet's cell **A1** into cell **B25** of the main worksheet

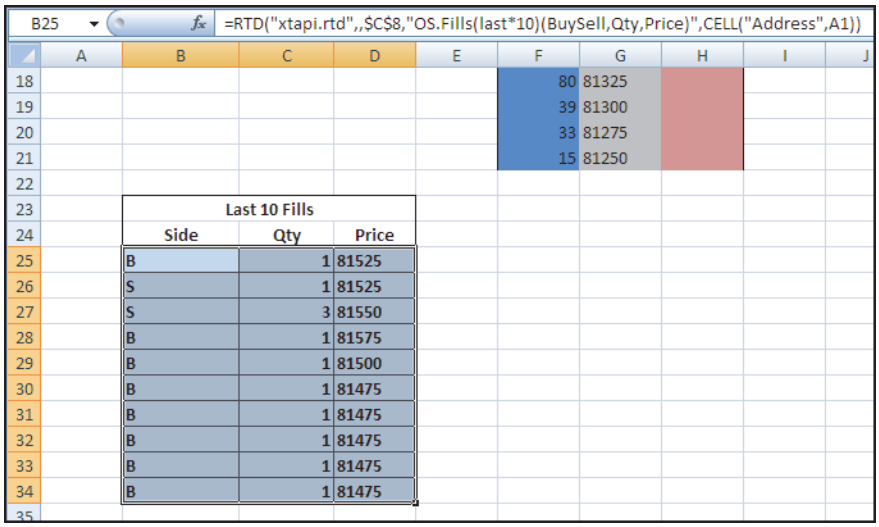
3. To display the side for the next nine fills, place the cursor in cell **B25** and select cells down through cell **B34**; then type **Ctrl+D** to fill-down (copy the formula to each of the selected cells).

   Assuming you have fills, the cells show the side of the market on which the fill occurred, similar to the following:



4. Unlike the price ladder, you are displaying the data in the same order it appears in the temporary spreadsheet. Therefore, you can take full advantage of Excel's fill-right feature. To finish the grid and include the **Qty** and **Price** data, select cells **B25**-**D34**; then type **Ctrl+R** to copy the formulas to the empty cells.

   When finished, the **Last 10 Fills** grid resembles the following.



5. If no fills appear, you can switch to X_TRADER® and generate some orders and fills so you can watch the grid update.

Creating the working orders grid

Overview

You can use the same technique to return information about the orders the trader currently has working in the market. Because working orders are also managed by an order set, you again use the **+OS** function, but specify the **Orders** method. In this tutorial, you want to track the first (oldest) working orders, so you can use the **first** keyword. To help get a picture of the working orders, you request the following order attributes:

- **BuySell**, which indicates the side of the market
- **Qty**, which indicates the number of contracts in this order
- **Stop**, which returns the price for a Stop order
- **Limit**, which returns the price for a Limit order

When you put all of these requirements together, you need to use the following formula:

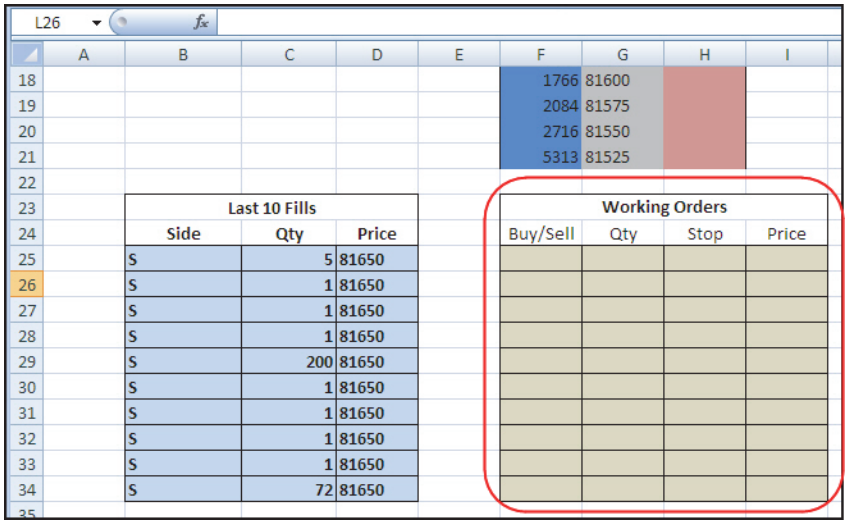
```
=RTD("xtapi.rtd",,$C$8,"OS.Orders(first*10)(BuySell,Qty,Stop,Limit)",CEL
L("Address",A1))
```

Defining and labeling the working orders grid

To help visualize the orders currently working in the market, you can label the grid area and assign different background colors for the columns.

To define and label the working orders grid:

1. Select cells **F23**-**I34**; then from the **Borders** menu, choose **Outside Borders**.
2. Set up the heading and grid rows as follows:



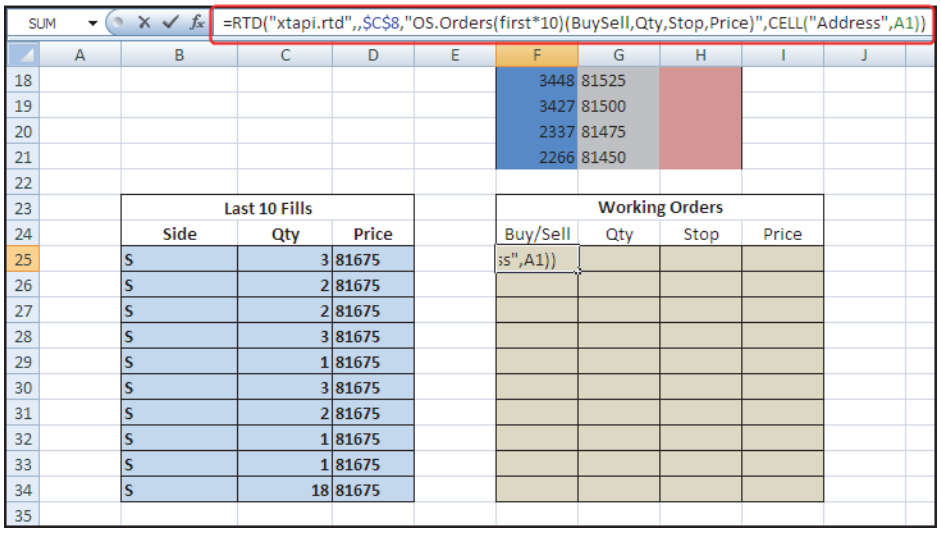
Adding the working order data formulas to the grid

Now that you have the market grid in place, you are ready to enter the RTD formulas to display the live market data.

To add the working order data formulas to the grid:

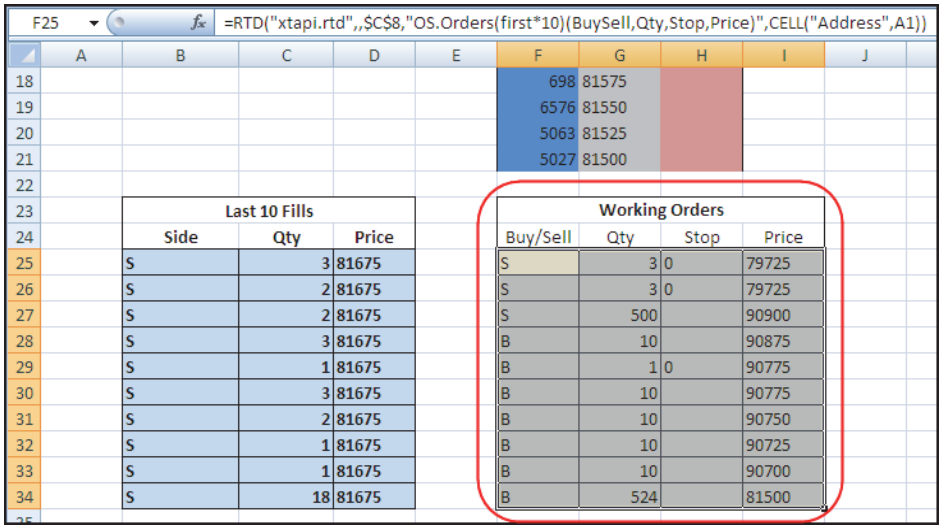
1. Starting with the **Buy/Sell** column, place the cursor in cell **F25**.
2. Enter the following formula:

```
=RTD("xtapi.rtd",,$C$8,"OS.Orders(last*10)(BuySell,Qty,Stop,Price)",CELL
("Address",A1))
```



3. To display the next nine fills, place the cursor in cell **F25** and select cells down through cell **F34**; then type **Ctrl+D** to fill-down (copy the formula to each of the selected cells).
4. To finish the grid and include the **Qty**, **Stop**, and **Price** data, select cells **F25**-**I34**; then type **Ctrl+R** to copy the formulas to the empty cells.

   When finished, the **Working Orders** grid resembles the following.



5. If no working orders appear, you can switch to X_TRADER® and generate some orders so you can watch the grid update.

Testing your application

Exploring market conditions

To test your application:

1. Enter qualifiers for a valid contract.

   The spreadsheet updates the price and market data fields to show current market conditions.
2. In X_TRADER®, submit Buy and Sell orders in the inside market for the same contract.

   By submitting orders on both sides of the market, you ensure that working orders and fills exist for the XTAPI RTD Server to display in the spreadsheet.
3. Observe the changes in the spreadsheet for the market data and the **Last 10 Fills** and **Working Orders** grids.

Tip: If you want to verify the accuracy of the data, you can open an X_TRADER® Order Book window and compare the values.

Saving your work

Before continuing to the next chapter, save your work and exit the Excel application.

6 Enhancing the Application

Chapter overview

This chapter walks you through the process of putting the final touches on the market monitoring application. It shows you how to access additional market data and order statuses. It also explains how to convert retrieved data so that the values use the same data types.

In this chapter

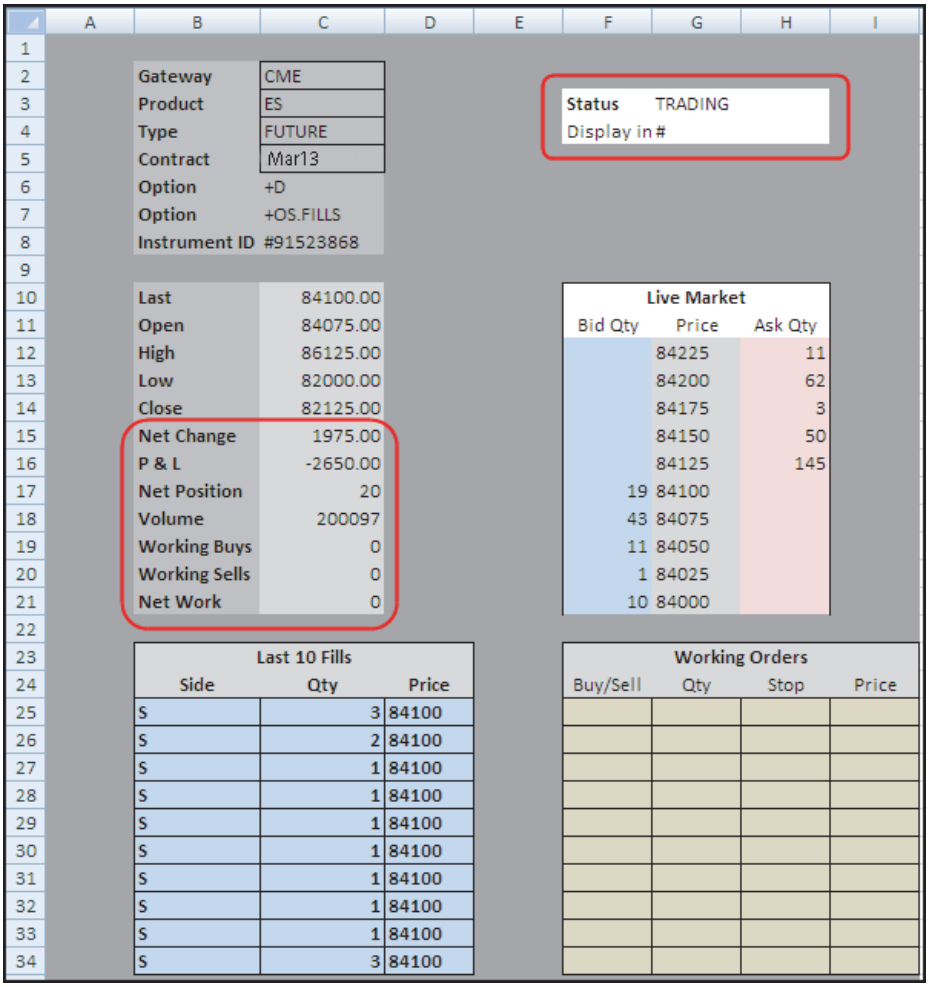
Overview

In review

In the last chapter, you expanded your application to include lists of fills and working orders to go along with the basic market data for the instrument.

Displaying additional instrument information

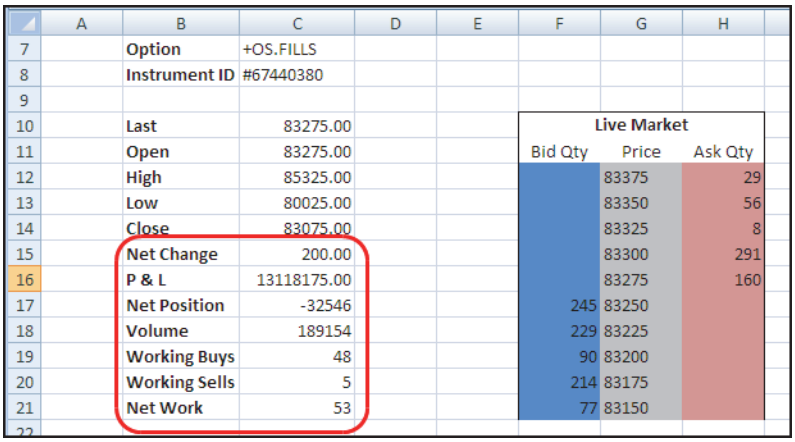
With the addition of fills and orders, your application can now access information from those fills and orders to give you a more complete picture of your current position and risk, such as your net change, position, and P&L. You can also customize how you want to display the information. In this chapter, you add two sections to the application to support this functionality, as shown in the following illustration.



Displaying additional market data

Overview

To provide a more complete view of the market, you can get more instrument data. In this chapter, you update the market data section to include additional information about the instrument, as shown.



	A	B	C	D	E	F	G	H
7		Option	+OS.FILLS					
8		Instrument ID	#67440380					
9								
10		Last	83275.00			Live Market		
11		Open	83275.00			Bid Qty	Price	Ask Qty
12		High	85325.00				83375	29
13		Low	80025.00				83350	56
14		Close	83075.00				83325	8
15		Net Change	200.00				83300	291
16		P & L	13118175.00				83275	160
17		Net Position	-32546			245	83250	
18		Volume	189154			229	83225	
19		Working Buys	48			90	83200	
20		Working Sells	5			214	83175	
21		Net Work	53			77	83150	
22								

When the display name differs from the attribute name

When you added formulas for the earlier market data attributes, you used the label text as the attribute to retrieve. For example, the **Last** label in cell **B10** matched the name of the attribute for the last price, so you used it in the formula in cell **C10**. The labels for the new market data are clear to a trader, but do not match the corresponding instrument attributes. You could continue naming labels that match, but might end up with some labels that aren't quite as clear.

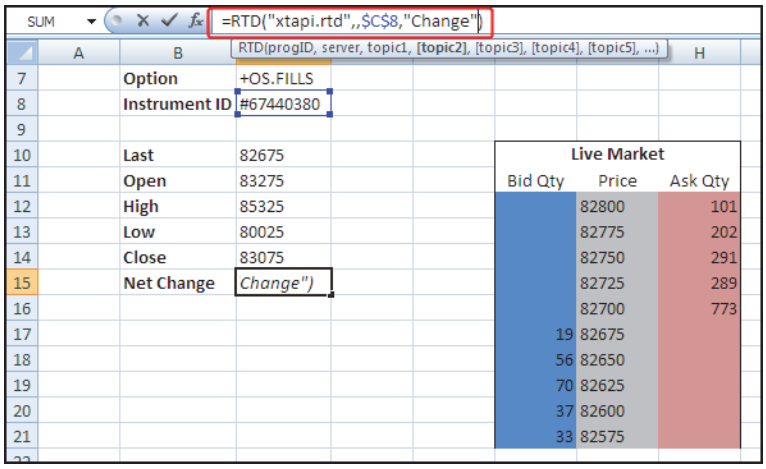
For example, to display the net change for an instrument, you could use the attribute name, **Change**, but "Change" is somewhat ambiguous and might cause confusion. For these new attributes, the tutorial keeps the labels and data lookup formulas separate. For example, the following formula retrieves the net change in price for the instrument.

```
=RTD("xtapi.rtd",,$C$8,"Change")
```

Updating the spreadsheet

To update the spreadsheet with the new market data values:

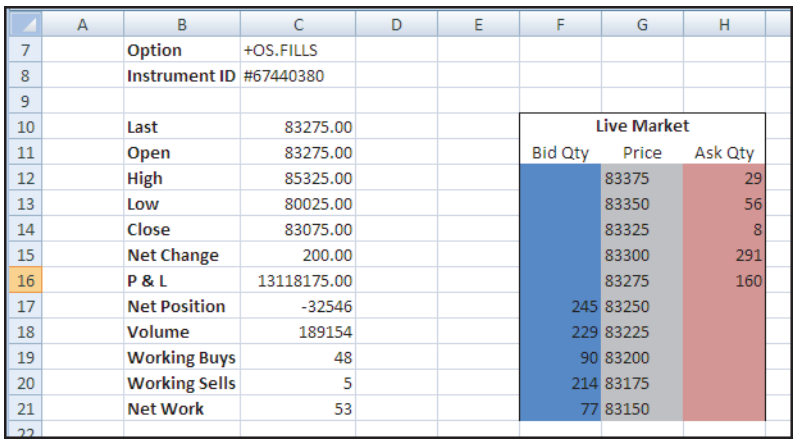
1. Open the spreadsheet, if necessary.
2. In cell **B15**, enter **Net Change** and format it as bold text.
3. Place the cursor in cell **C15** and enter the formula as shown.



4. Complete cells **B16**-**C21** using the following labels and formulas.

P&L	=RTD("xtapi.rtd",,C8,"PL")
Net Position	=RTD("xtapi.rtd",,C8,"NetPos")
Volume	=RTD("xtapi.rtd",,C8,"Volume")
Working Buys	=RTD("xtapi.rtd",,C8,"BuyWrk")
Working Sells	=RTD("xtapi.rtd",,C8,"SellWrk")
Net Work	=RTD("xtapi.rtd",,C8,"NetWrk")

When finished, your spreadsheet should resemble the following.

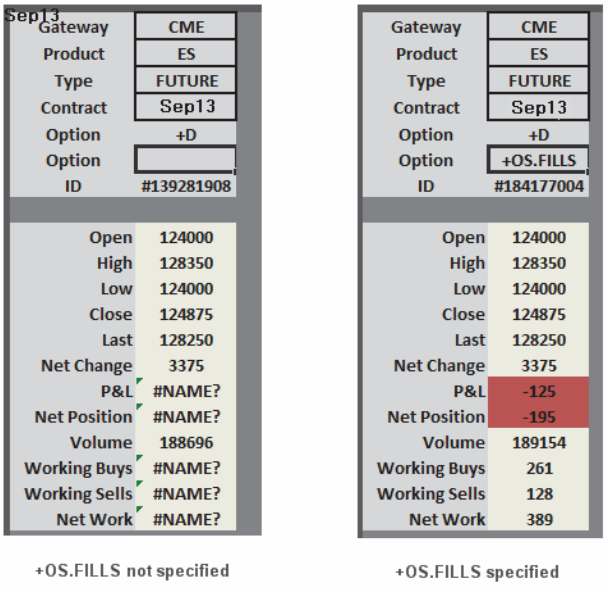


5. Save the spreadsheet.

An Excel error you might commonly see

Many of the instrument attributes are special cases because the XTAPI must inspect the trader's fills before it can properly make a calculation and return a value. For example, the calculation for P&L requires fill information, which is available only if the order set enables fill retention. You already enabled this functionality when you added the **+OS.FILLS** option to the **Instrument ID** formula. However, if you had not included this option, Excel would have displayed an error for these values.

The following snapshots illustrate how this option affects the values displayed in the spreadsheet.



As you can see, explicitly enabling the **+OS.FILLS** option allows the XTAPI RTD Server to populate the cells with valid values instead of errors.

Note: In an XTAPI RTD spreadsheet, any attributes that relate to positions or to profit and loss return Excel errors unless the order set retains fills.

Showing the contract trading status

Overview

While monitoring session activity for an instrument, you might want to know whether the selected contract is currently trading. So, now you will add a field to display the current trading status of the contract on the exchange (such as, TRADING, PRE-TRADING, CLOSED, EXPIRED, and so on). The

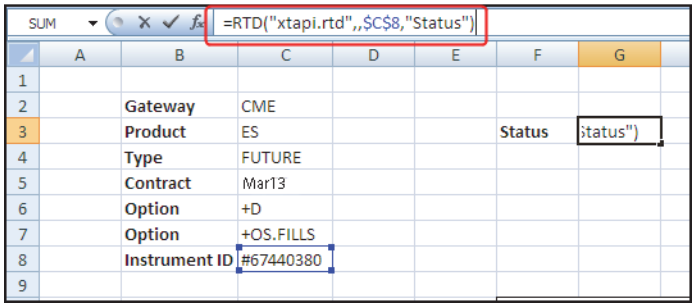
Status

attribute supplies this value.

Adding the trading status

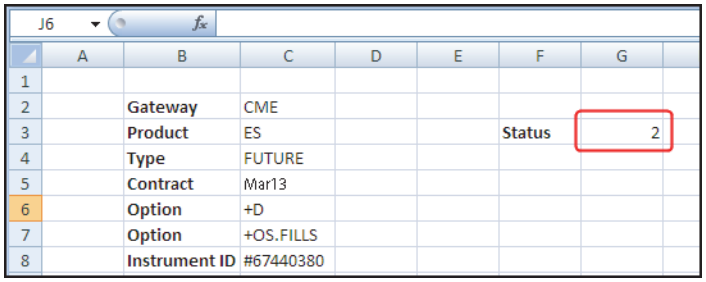
To add the trading status to the spreadsheet:

1. Open the spreadsheet, if necessary.
2. In cell **F3**, enter **Status** and format as bold text.
3. In cell **G3**, enter the formula as shown.



The spreadsheet updates, similar to the following.

4. Observe the contents of cell **G3**.

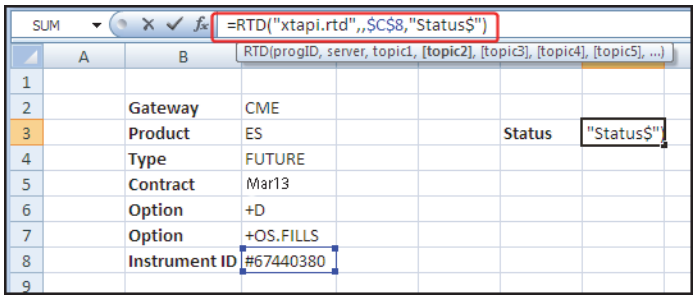


As you can see, the XTAPI RTD Server populated the cell with a number instead of the expected status. By default, the XTAPI formats the

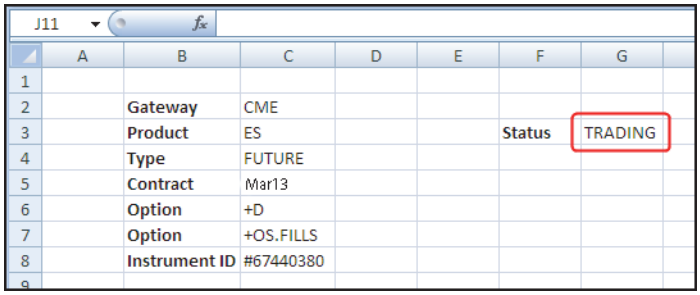
Status

attribute as an integer. You can use the $ modifier to instruct the XTAPI to return the value as a string.

5. In cell **G3**, modify the formula as follows.



6. Notice that the value changes to the text string that corresponds to the status integer.



Note: Many instrument attributes support different types of return types, such as decimal, string, X_TRADER® display format, and others. For information about attribute return types, refer to the

XTAPI Class Reference

.

7. Save the spreadsheet.

Normalizing values for calculations

Overview

As you saw in the last procedure, XTAPI can return values in a variety of data types and formats and that in some cases, you can specify which data type you want. If you only want to display data in your spreadsheet, you might not be concerned with whether the XTAPI RTD Server returns a value as an integer, decimal, or string. However, if you plan to use values to perform other calculations, you should make sure all of the values have the same output type to avoid calculation errors.

For example, suppose you display the Last Traded Price (**Last** attribute) in decimal format and the Opening Price (**Open** attribute) as an integer (ticks). If you later create a formula that subtracts one from the other, the result would be a meaningless value. This procedure shows you how to normalize the output types so that you can perform accurate calculations.

Output type indicators

The XTAPI supports several different output types to allow you to work with its data as you want. To specify an output type, you append an attribute name with one of the following symbols:

- **$**: string
- **#**: decimal
- **&**: ticks

When performing mathematical calculations, TT recommends that you use the decimal or tick formats.

Note: Not all attributes support all data types. Refer to the *XTAPI RTD Server Feature* to determine which output types you can use for a particular attribute.

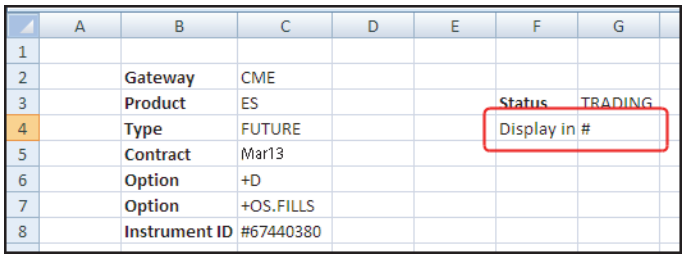
Specifying the output type indicator

For flexibility, you want to give the trader the ability to change the output types as desired. You can use the same approach you used for specifying basic attributes; namely, referencing the contents of a cell.

To specify an output type indicator:

1. Open the spreadsheet, if necessary.
2. In cell **F4**, enter **Display in** and format as desired.
3. To set the default output type to Decimal, in cell **G4** enter **#**.

   When finished, the spreadsheet should resemble the following.



	A	B	C	D	E	F	G
1							
2		Gateway	CME				
3		Product	ES			Status	TRADING
4		Type	FUTURE			Display in	#
5		Contract	Mar13				
6		Option	+D				
7		Option	+OS.FILLS				
8		Instrument ID	#67440380				

Adjusting attribute formulas to use the output type

Now that you provided a way for a trader to specify an output type, you need to adjust the formulas so that they convert the return values to the specified type. As a reminder, you append the indicator to the attribute name to convert the data

type. For example, to return the value of **Last** as a decimal value, you specify **Last#** as the attribute name, as in:

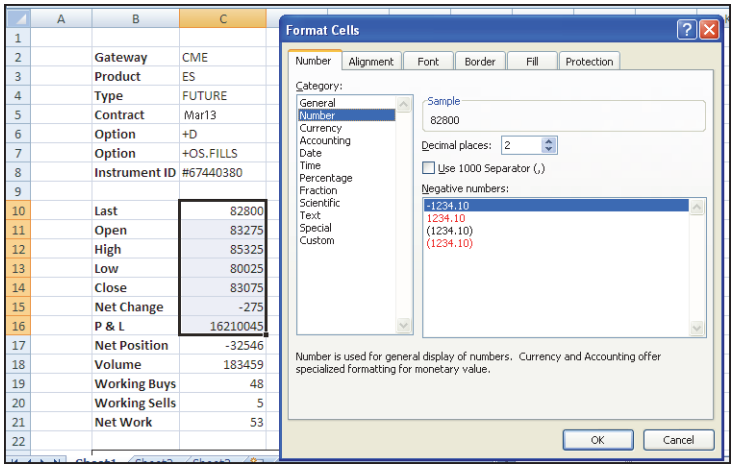
```
=RTD("xtaip.rtd",,$C$8,"Last#")
```

Fortunately, Excel provides a simple way to join the attribute name and the indicator, so you don't have to change the labels or hard-code the output types in the formulas. The Excel "**&**" operator concatenates two values, and you can use this to concatenate the indicator in cell **G4** with the attribute name or cell reference in each of the formulas, as in:

```
=RTD("xtaip.rtd",,$C$8,B10&G4)
=RTD("xtaip.rtd",,$C$8,"PL"&G4)
```

To adjust the attribute formulas to use the output type:

1. Open the spreadsheet, if necessary.
2. To make sure the spreadsheet displays the values as decimals, you need to change the cell formats.
   1 Select cells **C11**-**C16**.
   2 Open the **Cells Format** dialog.



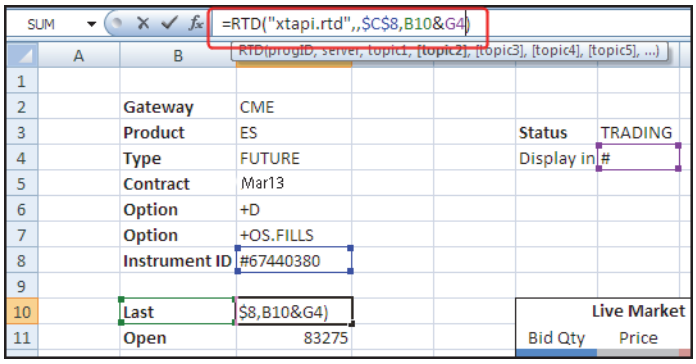
	A	B	C
1			
2		Gateway	CME
3		Product	ES
4		Type	FUTURE
5		Contract	Mar13
6		Option	+D
7		Option	+OS.FILLS
8		Instrument ID	#67440380
9			
10		Last	82800
11		Open	83275
12		High	85325
13		Low	80025
14		Close	83075
15		Net Change	-275
16		P & L	16210045
17		Net Position	-32546
18		Volume	183459
19		Working Buys	48
20		Working Sells	5
21		Net Work	53
22			

   3 In the **Category** list, select **Number**; then click **OK**.

   The Excel display format updates as follows.

9			
10		Last	83125
11		Open	83275
12		High	85325
13		Low	80025
14		Close	83075
15		Net Change	50
16		P & L	14094555.00
17		Net Position	-32546
18		Volume	185103
19		Working Buys	48
20		Working Sells	5
21		Net Work	53

3. In cell **C10**, add the **G4** cell reference to the **B10** reference in the formula as follows:



4. Repeat the process for cells **C11**-**C14**, all of which use cell references in the formulas.
5. In cells **C15** and **C16**, adjust the formulas to appending cell **G4** to the attribute name string, as shown:

```
=RTD("xtapi.rtd",,$C$8,"Change"&G4)
```

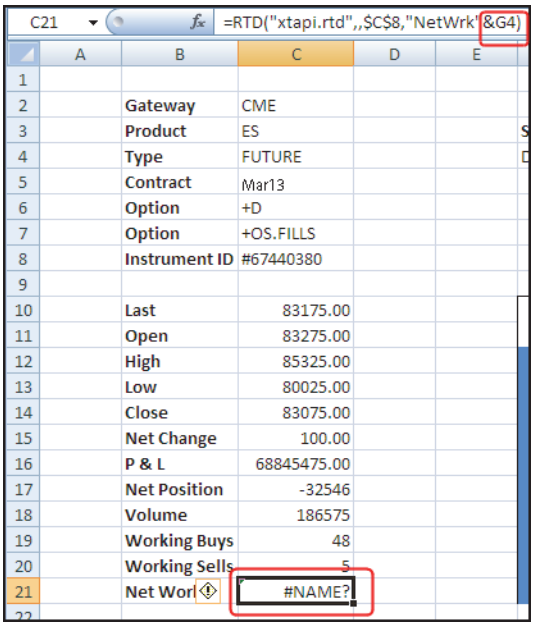
When finished, the attributes cells resemble the following.

10	Last	83150.00
11	Open	83275.00
12	High	85325.00
13	Low	80025.00
14	Close	83075.00
15	Net Change	75.00
16	P & L	13931825.00
17	Net Position	-32546

An Excel error you might occasionally see

As mentioned previously, the XTAPI does not necessarily support all output types for all attributes. The concatenation operator works in your spreadsheet because the resulting attribute name string is explicitly supported for that attribute. If you try to append an output operator to an attribute that it doesn't support, Excel displays an error to indicate that the XTAPI RTD Server could not find that attribute.

The following example shows the error that occurs if you try to request the **NetWork#** attribute, because the XTAPI does not support the decimal output for this quantity.



Note: If you receive this error when using output indicators, refer to the *XTAPI RTD Feature Guide* to determine whether it supports that output type.

Testing your application

Exploring market conditions

To test your application:

1. Enter qualifiers for a valid contract.

   The spreadsheet updates the price and market data fields to show current market conditions.
2. In X_TRADER®, submit Buy and Sell orders in the inside market for the same contract.

   By submitting orders on both sides of the market, you ensure that working orders and fills exist for the XTAPI RTD Server to display in the spreadsheet.
3. Observe the changes in the spreadsheet for the market data you added in this chapter, such as **Volume**, **Working Buys**, and so on.
4. Change the **Display in** value to **$**, **#**, and **&** to see how the price display changes based on the selected display format.

Tip: If you want to verify the accuracy of the data, you can open an X_TRADER® Order Book window and compare the values.

Saving your work

Before continuing to the next chapter, save your work and exit the Excel application.

7 Creating a Global Fill Book

Chapter overview

In this chapter, you expand the functionality of the spreadsheet so that it can handle multiple contracts. It focuses on the functionality you can access through an order set by displaying fill information for all contracts trading through an account.

In this chapter

Overview

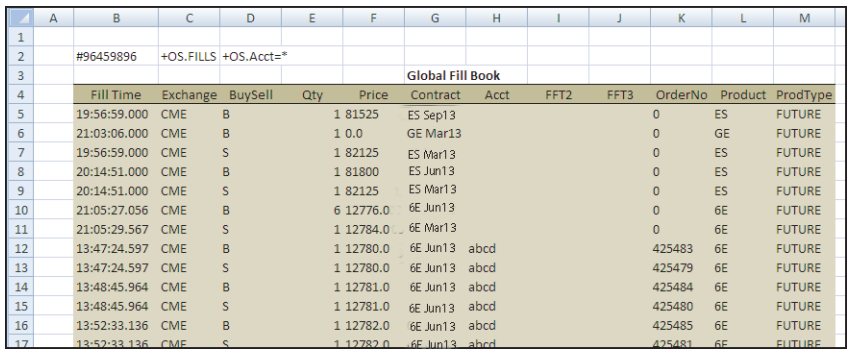
In review

So far, this tutorial focused on a single worksheet within an Excel spreadsheet. For the market monitoring functionality, this approach worked well, because everything within the spreadsheet was linked to the Instrument ID in cell **C8**. Changing the **Exchange**, **Product**, **Contract**, or **Product Type** values would change the entire spreadsheet to key off the new instrument.

This approach, however, does limit the spreadsheet's functionality, as it can display information only for the single contract specified in cell **C8**. If you trade multiple contracts, for example, you cannot display the fills for the other contracts.

Your goal for this chapter

In this chapter, you mimic the X_TRADER® **Fill Window** by creating a new worksheet within your spreadsheet that displays fills for all contracts associated with a trading account, similar to the following.



	A	B	C	D	E	F	G	H	I	J	K	L	M
1													
2		#96459896	+OS.FILLS	+OS.Acct=*									
3							Global Fill Book						
4		Fill Time	Exchange	BuySell	Qty	Price	Contract	Acct	FFT2	FFT3	OrderNo	Product	ProdType
5		19:56:59.000	CME	B	1	81525	ES Sep13				0	ES	FUTURE
6		21:03:06.000	CME	B	1	0.0	GE Mar13				0	GE	FUTURE
7		19:56:59.000	CME	S	1	82125	ES Mar13				0	ES	FUTURE
8		20:14:51.000	CME	B	1	81800	ES Jun13				0	ES	FUTURE
9		20:14:51.000	CME	S	1	82125	ES Mar13				0	ES	FUTURE
10		21:05:27.056	CME	B	6	12776.0	6E Jun13				0	6E	FUTURE
11		21:05:29.567	CME	S	1	12784.0	6E Mar13				0	6E	FUTURE
12		13:47:24.597	CME	B	1	12780.0	6E Jun13	abcd			425483	6E	FUTURE
13		13:47:24.597	CME	S	1	12780.0	6E Jun13	abcd			425479	6E	FUTURE
14		13:48:45.964	CME	B	1	12781.0	6E Jun13	abcd			425484	6E	FUTURE
15		13:48:45.964	CME	S	1	12781.0	6E Jun13	abcd			425480	6E	FUTURE
16		13:52:33.136	CME	B	1	12782.0	6E Jun13	abcd			425485	6E	FUTURE
17		13:52:33.136	CME	S	1	12782.0	6E Jun13	abcd			425481	6E	FUTURE

Adding another worksheet

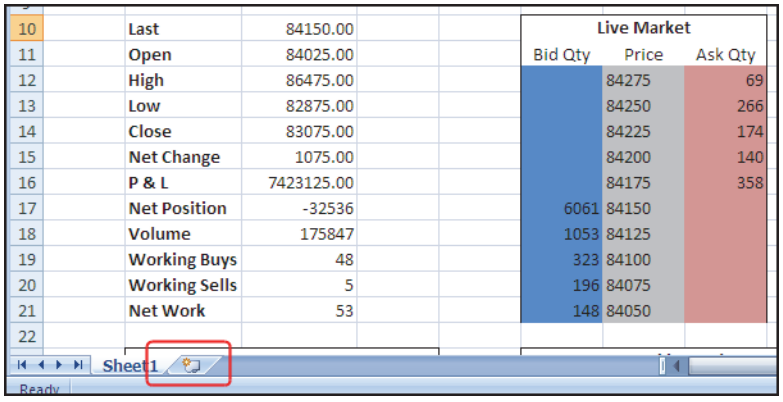
Why a new worksheet?

While you could add the fill book functionality to the main worksheet, creating a separate worksheet visually separates the different functionality and helps keep the information visible on the screen display.

Adding the worksheet

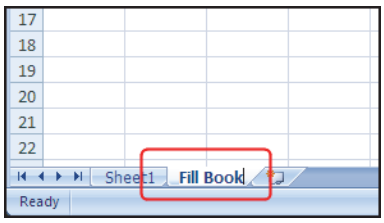
To add a new worksheet to the Excel spreadsheet:

1. Open the spreadsheet, if necessary.
2. In the bottom-left corner of the spreadsheet, click the **Insert Worksheet** tab, as shown.



Excel adds a new tab named **Sheet2**.

3. Double-click **Sheet2** and change the tab name to **Fill Book**, as follows.



4. Save the spreadsheet.

Specifying an order set

Overview

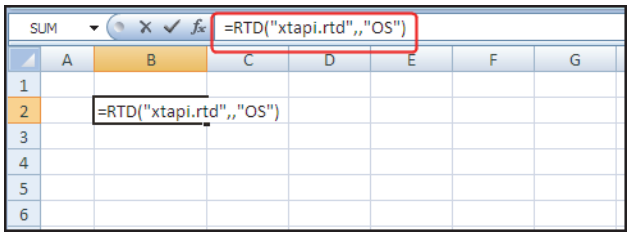
In the other worksheet, you asked the XTAPI RTD Server to create an Instrument ID that uniquely identified an instrument so that you could easily extract information using the ID. When working with fills in a fill book, you can use an ID to reference an order set in the RTD formulas. By creating a new Order Set, you can specify the parameters of any new or existing order that should be a part of this order set. Similar to the Instrument specification, this Order Set formula returns an ID that you can reference in RTD formulas.

Creating an order set ID

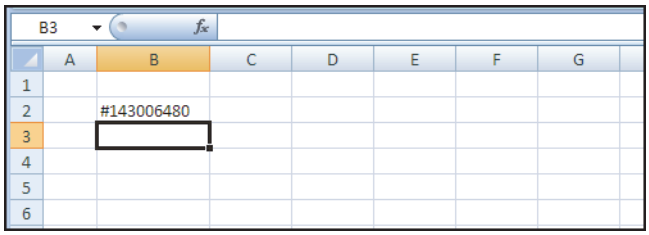
The **OS** option in an `RTD()` function call causes the XTAPI RTD Server to generate an order set ID.

To create an order set ID:

1. Open the spreadsheet, if necessary, and make sure the **Fill Book** tab is active.
2. In cell **B2**, enter the following formula:



The XTAPI RTD Server fills the cell with an ID, as shown.



3. Save the worksheet.

Adding order set options

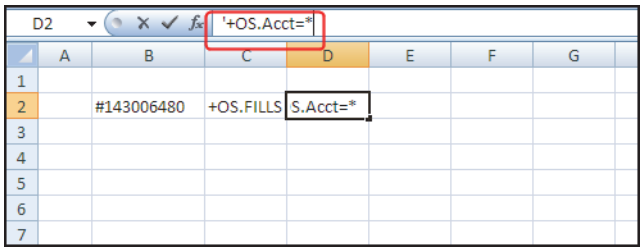
Now you have an Order Set ID that allows you to retrieve information from the order set. However, you still need to enable the order set to retain fill records and to provide visibility to all contracts for an account.

To add order set options:

1. In cell **C2**, specify that you want the order set to retain fill records by entering the following. Just as you did in previous exercises, make sure you use the single quote.

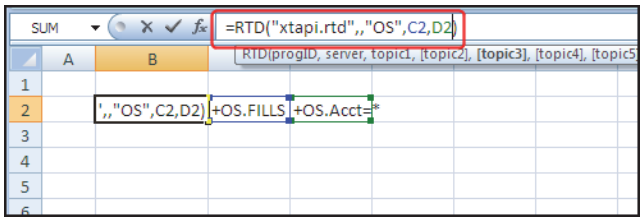
```
'+OS.FILLS
```

2. In cell **D2**, specify the account number as follows.



Note: The asterisk argument tells the XTAPI RTD Server to monitor all accounts for which Order Book sharing is enabled. You can replace the asterisk in the example with your actual account ID to restrict the display to all contracts for your account.

3. Modify the order set ID function in cell **B2** to use these new options as follows:



4. Save the spreadsheet.

Creating the fill book display

What to include in the fill book

In the same way you can access Instrument attributes through the XTAPI RTD Server, you can also retrieve the values of order set attributes. For this tutorial, you subscribe to a small subset of the possible attributes, including:

Attribute name	Contents
TimeExec	Time the fill occurred
Exchange	ID of the exchange that generated the fill
BuySell	Side of the market of the fill
Qty	Number of contracts filled
Price	Price at which the fill occurred
Contract	Contract associated with the filled order
Acct	Trader account number for the order and fill
FFT2 and FFT3	Optional information included with the fill
OrderNo	Number for the order that matches the fill
Product	Name of the product in the fill
ProdType	Type of product filled

For a complete list of order set attributes, refer to the **`TTOrderSet`** object in the *XTAPI RTD Feature Guide*.

Accessing the fill attributes

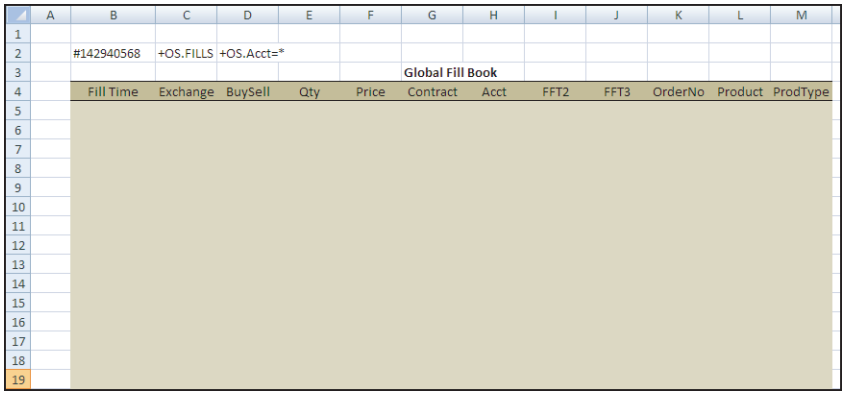
Earlier in the tutorial, you accessed the **BuySell**, **Qty**, and **Price** fill attributes to display in your **Last 10 Fills** grid. In that situation, you asked the XTAPI RTD Server to get the attributes of the fills associated with the selected instrument. Because fills result from orders submitted through an order set, the XTAPI tracks fills through an order set, not directly through an instrument. Because you accessed the fill information through an instrument, you had to use **OS.Fills** in the `RTD()` function call to instruct the XTAPI to look in the order set instead of the instrument for the fill attributes.

In the Global Fill Book you are now developing, you return an Order Set ID, not an Instrument ID. Therefore, you can reference the fill attributes directly through the order set whose ID is stored in cell **B2**. When calling the `RTD()` function with the order set ID, you specify **Fills** instead of **OS.Fills** for the parameter.

Creating the fill book grid

To create the fill book grid:

1. Open the spreadsheet, if necessary, and make sure the **Fill Book** worksheet is visible.
2. In row 4, add the **Global Fill Book** title. Position and format as desired.
3. In cells **B5**-**M5**, enter column headings for each attribute similar to the following:



4. If you prefer, you can change the background color and cell ruling for the heading row and table body to suit your taste.
5. Save the spreadsheet.

Creating the fill book formula

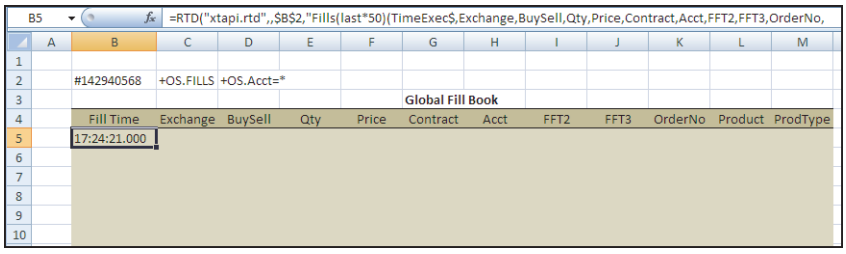
As you want the XTAPI RTD Server to return several different variables, you should realize that you can use the temporary, hidden worksheet (as described in the section called, Retrieving the data from the array) to store the data. You can then just insert that hidden worksheet into the appropriate cells in the **Global Fill Book** grid. In this tutorial, you display the 50 most-recent fills, though you could display any number of them. By default, the XTAPI RTD Server sorts the fills in date/time order, with the newest fills shown first.

To create the fill book formula:

1. Place the cursor in cell **B5**.
2. Enter the following formula:

```
=RTD("xtapi.rtd",,$B$2,"Fills(last*50)(TimeExec$,Exchange,BuySell,
Qty,Price,Contract,Acct,FFT2,FFT3,OrderNo,Contract.Product,
Contract.ProdType,)", CELL("Address",A1))
```

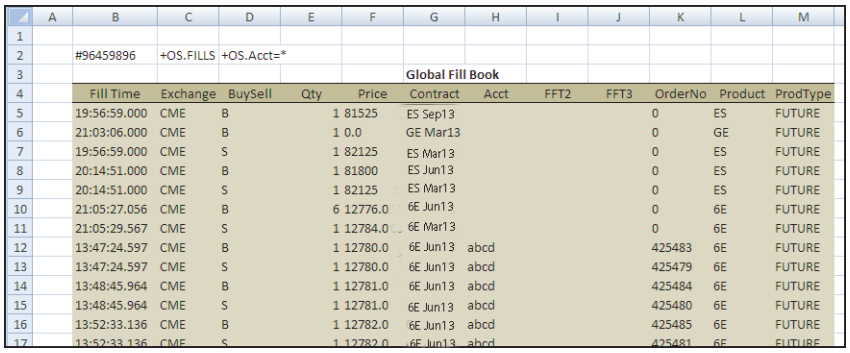
The **Global Fill Book** grid populates the time of the first fill, similar to the following.



Tip: The **Product** and **ProdType** attributes are not attributes of the order set itself, but of the instrument in the order set. To access these values, you need to specify **Contract.Product** and **Contract.ProdType** as the attributes to tell the XTAPI RTD Server to get the values from the Instrument object instead.

3. Select cells **B5**-**M55**; then use Excel's fill-right and fill-down commands to populate the entire grid.

   The rest of the grid populates, as shown.



	A	B	C	D	E	F	G	H	I	J	K	L	M
1													
2		#96459896	+OS.FILLS	+OS.Acct=*									
3							Global Fill Book						
4		Fill Time	Exchange	BuySell	Qty	Price	Contract	Acct	FFT2	FFT3	OrderNo	Product	ProdType
5		19:56:59.000	CME	B	1	81525	ES Sep13				0	ES	FUTURE
6		21:03:06.000	CME	B	1	0.0	GE Mar13				0	GE	FUTURE
7		19:56:59.000	CME	S	1	82125	ES Mar13				0	ES	FUTURE
8		20:14:51.000	CME	B	1	81800	ES Jun13				0	ES	FUTURE
9		20:14:51.000	CME	S	1	82125	ES Mar13				0	ES	FUTURE
10		21:05:27.056	CME	B	6	12776.0	6E Jun13				0	6E	FUTURE
11		21:05:29.567	CME	S	1	12784.0	6E Mar13				0	6E	FUTURE
12		13:47:24.597	CME	B	1	12780.0	6E Jun13	abcd			425483	6E	FUTURE
13		13:47:24.597	CME	S	1	12780.0	6E Jun13	abcd			425479	6E	FUTURE
14		13:48:45.964	CME	B	1	12781.0	6E Jun13	abcd			425484	6E	FUTURE
15		13:48:45.964	CME	S	1	12781.0	6E Jun13	abcd			425480	6E	FUTURE
16		13:52:33.136	CME	B	1	12782.0	6E Jun13	abcd			425485	6E	FUTURE
17		13:52:33.136	CME	S	1	12782.0	6E Jun13	abcd			425481	6E	FUTURE

4. Save the spreadsheet.

Testing your application

Monitoring your fill book

To test your application:

1. In X_TRADER®, submit Buy and Sell orders in the inside market for the several different contracts.
2. In your RTD application, make the **Fill Book** tab active.
3. Observe the changes in the spreadsheet for the market data and the **Live Market** grid.

Tip: If you want to verify the accuracy of the data, you can open an X_TRADER® Fill Window and compare the values.

Saving your work

Before continuing to the next chapter, save your work and exit the Excel application.

This page intentionally left blank for printing purposes

8 Monitoring Time and Sales Data

Chapter overview

In this chapter, you expand the functionality of the spreadsheet so that it can monitor time and sales data for a contract.

Note: You can only access time and sales information if you use X_TRADER® API 7.7.8 or higher.

In this chapter

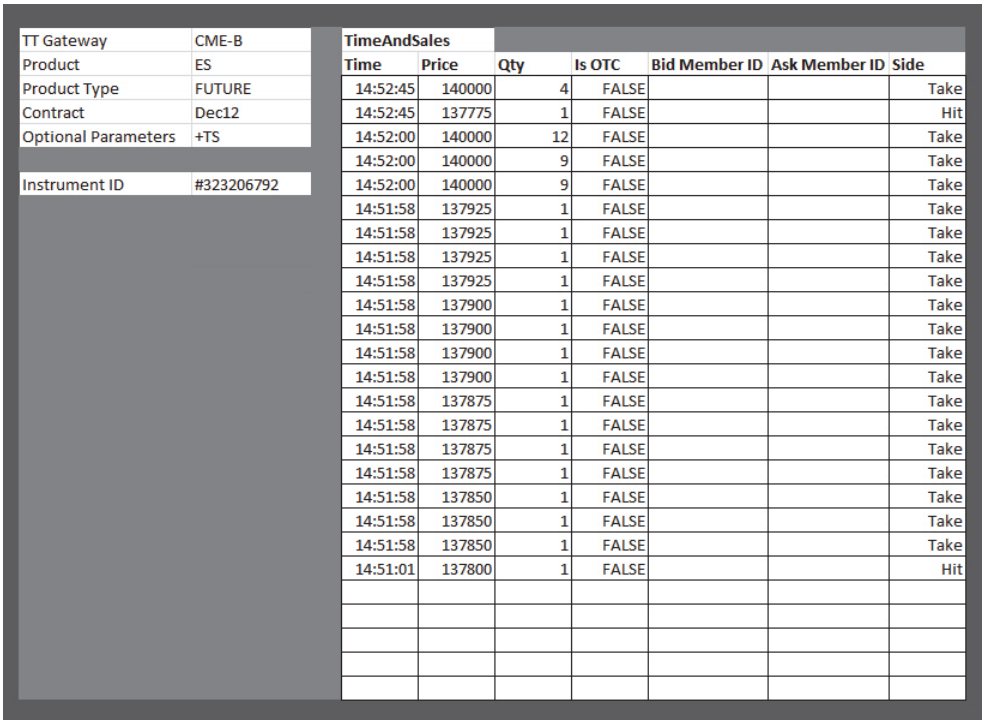
Overview

In review

To this point in the tutorial, you created a spreadsheet with two worksheets: one that displays live market data for a contract and one that displays fills for all contracts associated with a trading account. You learned to use the hidden spreadsheet feature in Excel to extract the value of a single instrument attribute.

Your goal for this chapter

In this chapter, you use the a hidden spreadsheet again when requesting time & sales data for a contract, as shown.



TT Gateway	CME-B
Product	ES
Product Type	FUTURE
Contract	Dec12
Optional Parameters	+TS
Instrument ID	#323206792

TimeAndSales

Time	Price	Qty	Is OTC	Bid Member ID	Ask Member ID	Side
14:52:45	140000	4	FALSE			Take
14:52:45	137775	1	FALSE			Hit
14:52:00	140000	12	FALSE			Take
14:52:00	140000	9	FALSE			Take
14:52:00	140000	9	FALSE			Take
14:51:58	137925	1	FALSE			Take
14:51:58	137925	1	FALSE			Take
14:51:58	137925	1	FALSE			Take
14:51:58	137925	1	FALSE			Take
14:51:58	137900	1	FALSE			Take
14:51:58	137900	1	FALSE			Take
14:51:58	137900	1	FALSE			Take
14:51:58	137900	1	FALSE			Take
14:51:58	137875	1	FALSE			Take
14:51:58	137875	1	FALSE			Take
14:51:58	137875	1	FALSE			Take
14:51:58	137875	1	FALSE			Take
14:51:58	137850	1	FALSE			Take
14:51:58	137850	1	FALSE			Take
14:51:58	137850	1	FALSE			Take
14:51:01	137800	1	FALSE			Hit

For more information about Time and Sales information, refer to the X_TRADER Help System.

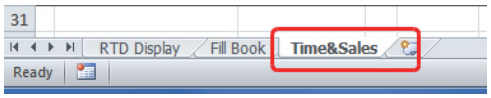
Adding another worksheet

Adding the worksheet

Just as you added a new worksheet for the Fill Book, you now add another new sheet for the time and sales data.

To add the Time&Sales worksheet to the Excel spreadsheet:

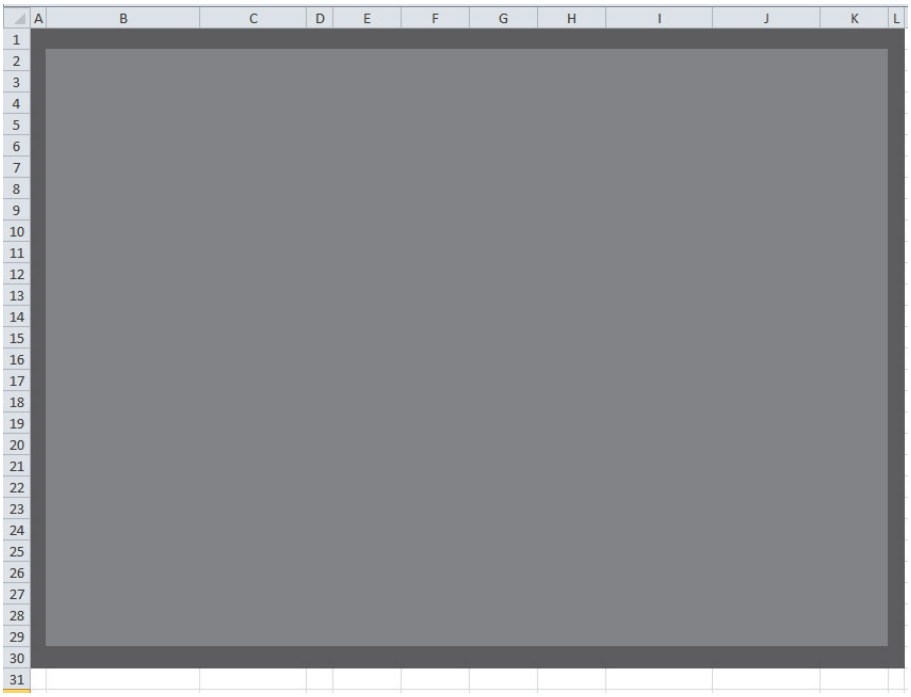
1. Open the spreadsheet, if necessary.
2. In the bottom-left corner of the spreadsheet, click the **Insert Worksheet** tab and name it **Time&Sales**, as shown.



3. Save the spreadsheet.

Organizing the layout

To help with the presentation of the data, you can set background colors for the spreadsheet, similar to the following.



Specifying the contract to monitor

Overview

In the **RTD Display** worksheet, you asked the XTAPI RTD Server to create an Instrument ID that uniquely identified an instrument so that you could easily extract information using the ID. You use a similar approach for this worksheet, you need to request time and sales information.

About time and sales data

As an optimization, the XTAPI RTD Server does not request time and sales information when accessing instrument data unless you specifically want it.

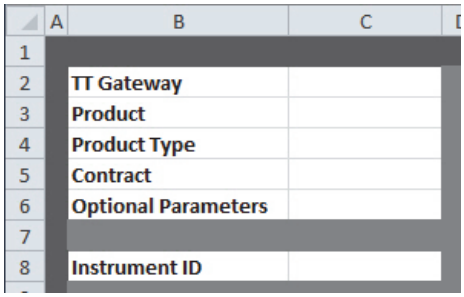
The XTAPI RTD Server allows you to pass options in the `RTD()` function call. The **+TS** option controls whether the XTAPI RTD Server requests and processes time and sales information.

Creating the contract section

First, you need to create a place for a trader to specify contract information. The tutorial application allows a trader to enter the **Gateway**, **Product**, **Type**, and **Contract** information for an instrument, as well as a place for **Optional Parameters**.

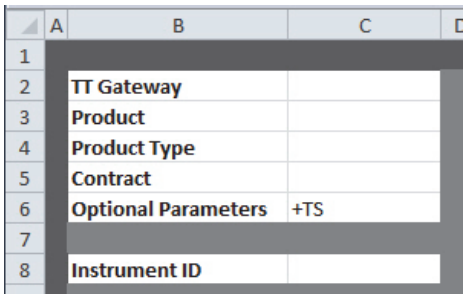
To add labels for the contract section:

1. Place the cursor in cell **B2**.
2. Type **TT Gateway**; then select the text and make it bold.
3. Repeat the process for cells **B3** through **B8** with the values: **Product**, **Type**, **Contract**, **Optional Parameters**, and **Instrument ID**, as follows.



As you can see, this section looks similar to the contract section you created in the **RTD Display** worksheet.

4. To enable time and sales data, place the cursor in cell **C6** and type **'+TS**. Remember, you need to use the quote (') to stop Excel from evaluating the expression.

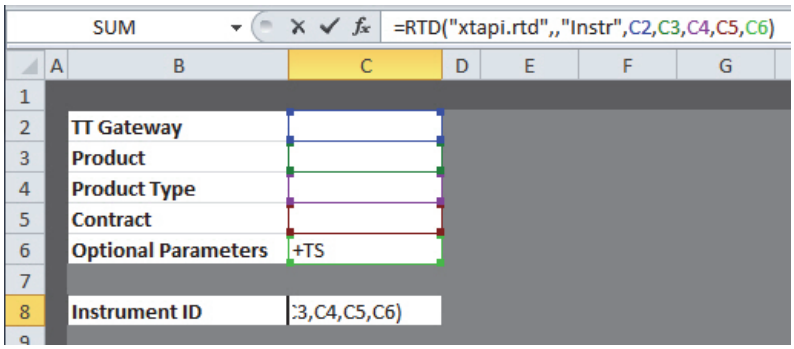


Generating an Instrument ID

Now that a trader can input contract qualifiers into the spreadsheet, you have the information to generate a unique ID that you can use to reference the instrument throughout the worksheet.

To generate an instrument ID for the specified contract qualifiers:

1. Place the cursor in cell **B8**.
2. Type **Instrument ID**; then select the text and make it bold. If necessary, resize the column to accommodate the text.
3. Place the cursor in cell **C8**.
4. Type: **=RTD("xtapi.rtd",,"Instr",C2,C3,C4,C5,C6)**, as shown.



When you enter the formula, Excel displays **#N/A** in the cell because the contract qualifier cells do not yet contain data.

Testing the Instrument ID

With the `RTD()` formula set to use the values from the input cells, you can now enter contract qualifiers to test your formula. After you enter values into each of the four qualifier cells, Excel updates the **Instrument ID** with an integer value.

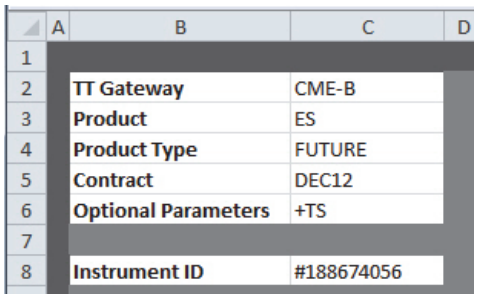
Note: You must precede the contract name with a single quote (') to prevent Excel from treating the value as a Date data type. If you enter the date without the quote, Excel applies its default display format for dates. For example, if you enter **Dec13**, Excel displays it as **13-Dec**. When you use the **Instrument ID** in another formula, Excel would pass **13-Dec** as the contract name. Consequently, the request would fail, as **13-Dec** does not represent a valid contract name.

You can also set the Excel cell format to **Text** to stop Excel from reformatting the date.

To test the Instrument ID formula:

1. Enter the following values. If you do not use the TT CME-B Gateway, choose alternate contract credentials.
   - Gateway: **CME-B**
   - Product: **ES**
   - Type: **FUTURE**
   - Contract: **'Dec13**

2. Observe that the **Instrument ID** field in cell **C8** now contains an integer similar to the following.



	A	B	C	D
1				
2		**TT Gateway**	CME-B	
3		**Product**	ES	
4		**Product Type**	FUTURE	
5		**Contract**	DEC12	
6		**Optional Parameters**	+TS	
7				
8		**Instrument ID**	#188674056	

Note: As a reminder, this value has no direct connection to the actual instrument, so you cannot rely on the value remaining the same each time you open the spreadsheet.

Creating the time and sales display

What time and sales data returns

When you use the **+TS** option to request time and sales data, the XTAPI RTD Server returns an array of values each time it detects a new trade event. The array contains the following values:

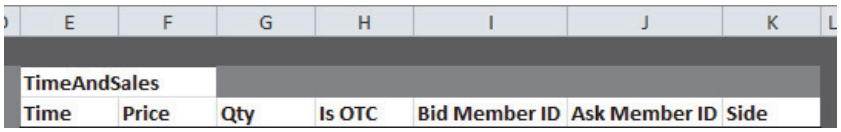
Value	Contents
Time	Time the trade occurred
Price	Price at which the trade occurred
Qty	Number of contracts traded
Is OTC	Whether the trade represents an over-the-counter transaction
Bid Member ID	ID of the member who placed the bid
Ask Member ID	ID of the member who placed the ask
Side	Side of the market that initiated the trade

Note: Not all exchanges provide values for all of these fields, so the RTD returns empty strings for the values.

Creating the time and sales grid

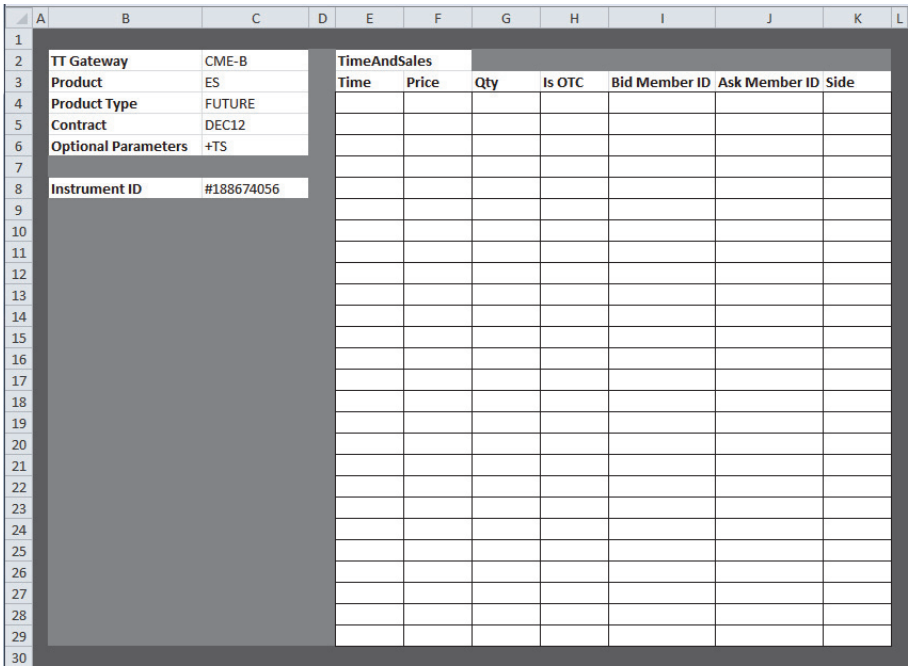
To create the time and sales grid:

1. Open the spreadsheet, if necessary, and make sure the **Time&Sales** worksheet is visible.
2. In cell **E2**, add the **TimeAndSales** title. Position and format as desired.
3. In cells **E3**-**K3**, enter column headings for each value similar to the following:



4. Select cells **E4**-**K29** (or adjust the number of rows to suit your needs) and display **All Borders** for the cells.

   Your spreadsheet should look similar to the following:



5. If you prefer, you can change the background color and cell ruling for the heading row and table body to suit your taste.
6. Save the spreadsheet.

Creating the time and sales formula

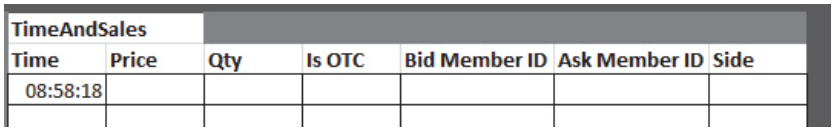
As you want the XTAPI RTD Server to return several different variables, you should realize that Excel uses the temporary, hidden worksheet (as described in the section called, Retrieving the data from the array) to store the data. As a result, you just insert that hidden worksheet into the appropriate cells in the **TimeAndSales** grid. In this tutorial, you display time and sales information for the 25 most-recent trades, though you could display any number of them. By default, the XTAPI RTD Server sorts the trades in date/time order, with the newest trades shown first.

▶ To create the time and sales formula:

1. Place the cursor in cell **E4**.
2. Enter the following formula:

```
=RTD("xtapi.rtd",,$C$8,$E$2,CELL("address",A1))
```

The **Global Fill Book** grid populates the time of the first fill, similar to the following.



3. Select cells **E5**-**K29**; then use Excel's fill-right and fill-down commands to populate the entire grid.

   The rest of the grid populates, as shown. As CME does not provide the Bid or Ask Member IDs, these fields are blank.

TimeAndSales						
Time	Price	Qty	Is OTC	Bid Member ID	Ask Member ID	Side
09:00:13	137850	10	FALSE			Take
09:00:12	137850	10	FALSE			Take
09:00:08	137825	5	FALSE			Hit
08:59:53	137825	1	FALSE			Hit
08:59:52	137850	10	FALSE			Take
08:59:52	137850	10	FALSE			Take
08:59:51	137850	17	FALSE			Take
08:59:06	137850	1	FALSE			Take
08:59:02	137850	3	FALSE			Take
08:58:59	137850	1	FALSE			Take
08:58:59	137850	3	FALSE			Take
08:58:57	137850	2	FALSE			Take
08:58:56	137850	1	FALSE			Take
08:58:55	137850	1	FALSE			Take
08:58:55	137850	1	FALSE			Take
08:58:54	137850	1	FALSE			Take
08:58:52	137850	1	FALSE			Take
08:58:51	137850	1	FALSE			Take
08:58:50	137850	1	FALSE			Take
08:58:50	137850	2	FALSE			Take
08:58:49	137850	2	FALSE			Take
08:58:49	137825	1	FALSE			Hit
08:58:49	137850	1	FALSE			Take
08:58:44	137850	5	FALSE			Take
08:58:44	137850	4	FALSE			Take
08:58:43	137850	4	FALSE			Take

4. Save the spreadsheet.

Testing your application

Monitoring your fill book

To test your application:

1. In X_TRADER®, submit Buy and Sell orders in the inside market for the several different contracts.
2. In your RTD application, make the **TimeAndSales** tab active.
3. Observe the changes in the spreadsheet for the time and sales data.

Saving your work

Before continuing to the next chapter, save your work and exit the Excel application.

Send Us Your Comments

X_TRADER® API RTD Tutorial

7.X

Trading Technologies® welcomes your comments and suggestions on the accuracy and usefulness of this publication. Your input is important and valuable in revising our documentation and helps ensure a constantly improving level of quality.

- Did you find any errors?
- Is the information clearly presented?
- Do you need more information? If so, where?
- Are the examples correct? Do you need more examples?
- Which features did you find particularly useful?
- What did you like most about this manual or document?

If you encounter any errors in this document or would like to share other suggestions you might have for improving this document, send comments to: documentation.dept@tradingtechnologies.com.

If possible, please indicate the chapter, section, and page number relevant to your feedback.