

Datafeed Toolbox

For Use with **MATLAB®**

- Computation
- Visualization
- Programming

User's Guide

Version 1



How to Contact The MathWorks



www.mathworks.com
comp.soft-sys.matlab
www.mathworks.com/contact_TS.html

Web
Newsgroup
Technical Support



suggest@mathworks.com
bugs@mathworks.com
doc@mathworks.com
service@mathworks.com
info@mathworks.com

Product enhancement suggestions
Bug reports
Documentation error reports
Order status, license renewals, passcodes
Sales, pricing, and general information



508-647-7000 (Phone)



508-647-7001 (Fax)



The MathWorks, Inc.
3 Apple Hill Drive
Natick, MA 01760-2098

For contact information about worldwide offices, see the MathWorks Web site.

Datafeed Toolbox User's Guide

© COPYRIGHT 1999–2006 by The MathWorks, Inc.

The software described in this document is furnished under a license agreement. The software may be used or copied only under the terms of the license agreement. No part of this manual may be photocopied or reproduced in any form without prior written consent from The MathWorks, Inc.

FEDERAL ACQUISITION: This provision applies to all acquisitions of the Program and Documentation by, for, or through the federal government of the United States. By accepting delivery of the Program or Documentation, the government hereby agrees that this software or documentation qualifies as commercial computer software or commercial computer software documentation as such terms are used or defined in FAR 12.212, DFARS Part 227.72, and DFARS 252.227-7014. Accordingly, the terms and conditions of this Agreement and only those rights specified in this Agreement, shall pertain to and govern the use, modification, reproduction, release, performance, display, and disclosure of the Program and Documentation by the federal government (or other entity acquiring for or through the federal government) and shall supersede any conflicting contractual terms or conditions. If this License fails to meet the government's needs or is inconsistent in any respect with federal procurement law, the government agrees to return the Program and Documentation, unused, to The MathWorks, Inc.

Trademarks

MATLAB, Simulink, Stateflow, Handle Graphics, Real-Time Workshop, and xPC TargetBox are registered trademarks, and SimBiology, SimEvents, and SimHydraulics are trademarks of The MathWorks, Inc.

Other product or brand names are trademarks or registered trademarks of their respective holders.

Patents

The MathWorks products are protected by one or more U.S. patents. Please see www.mathworks.com/patents for more information.

Revision History

December 1999	First printing	New for MATLAB 5.3 (Release 11)
June 2000	Online only	Revised for Version 1.2
December 2000	Online only	Revised for Version 1.3
February 2003	Online only	Revised for Version 1.4
June 2004	Online only	Revised for Version 1.5 (Release 14)
August 2004	Online only	Revised for Version 1.6 (Release 14+)
September 2005	Second printing	Revised for Version 1.7 (Release 14SP3)
March 2006	Online only	Revised for Version 1.8 (Release 2006a)
September 2006	Online only	Revised for Version 1.9 (Release 2006b)

Getting Started

1

What Is the Datafeed Toolbox?	1-2
Communicating with a Financial Data Server	1-3
Communication Management	1-4
Verifying the Connection	1-5
Retrieving Connection Properties	1-5
Disconnecting from a Data Server	1-6
Retrieving Data	1-7
Example: Retrieving Bloomberg Data	1-7
Datafeed Toolbox Graphical User Interface	1-15
Datafeed Dialog Box	1-15
Securities Lookup Dialog Box (Bloomberg, FT Interactive Data)	1-21

Functions — By Category

2

Bloomberg	2-2
Thomson Datastream	2-22
FactSet	2-30
Hyperfeed	2-39
FT Interactive Data	2-48

Federal Reserve Economic Data	2-55
Yahoo!	2-63

Functions — By Category

3

Bloomberg	3-2
Thomson Datastream	3-22
FactSet	3-30
Hyperfeed	3-39
FT Interactive Data	3-48
Federal Reserve Economic Data	3-55
Yahoo!	3-63

Related Information

A

Additional Software	A-2
Obtaining Client Software	A-2
Connecting to FactSet	A-2
Connecting to the Thomson Datastream API	A-2

Index

Getting Started

What Is the Datafeed Toolbox?
(p. 1-2)

Describes the purpose of the Datafeed Toolbox.

Communicating with a Financial Data Server (p. 1-3)

Describes how to establish communication with a financial data server.

Retrieving Data (p. 1-7)

Uses the Bloomberg `fetch` function to illustrate the steps involved in financial data retrieval.

Datafeed Toolbox Graphical User Interface (p. 1-15)

Illustrates the use of the graphical user interface `dftool` to obtain financial data from a data server.

What Is the Datafeed Toolbox?

This document describes the Datafeed Toolbox for MATLAB®. The Datafeed Toolbox effectively turns your MATLAB workstation into a financial data acquisition terminal. Using the Datafeed Toolbox, you can download a wide variety of security data from financial data servers into your MATLAB workspace. Then, you can pass this data to MATLAB or to another toolbox, such as the Financial Toolbox, for further analysis.

Communicating with a Financial Data Server

The Datafeed Toolbox supports connections to seven financial data servers, provided by the following corporations:

- Bloomberg L. P. (<http://www.bloomberg.com>)
- FactSet Research Systems, Inc. (<http://www.factset.com>)
- Hyperfeed Technologies, Inc. (<http://www.hyperfeed.com>)
- FT Interactive Data Corporation (<http://www.FTInteractiveData.com>)
- Thomson Corporation (<http://www.thomson.com>)
- Federal Reserve Economic Data (FRED®) (<http://research.stlouisfed.org/fred2/>)
- Yahoo!, Inc. (<http://www.yahoo.com>)

Bloomberg, FT Interactive Data, and Hyperfeed all require that you install proprietary software on your PC.

To connect to FactSet, Thomson Datastream, Federal Reserve Economic Data, or Yahoo!, you need access to the Internet. FactSet additionally requires that you be licensed to use FactSet's FAST technology.

There are four steps involved in communicating with a financial data server using this toolbox. They are

- 1** "Communication Management" on page 1-4
- 2** "Verifying the Connection" on page 1-5
- 3** "Retrieving Connection Properties" on page 1-5
- 4** "Disconnecting from a Data Server" on page 1-6

This chapter uses the Bloomberg financial data server as an example of how to establish communication with a financial data server and retrieve data. Communication with other supported data servers is accomplished with a virtually identical set of toolbox functions.

Communication Management

For each of the supported financial data servers, the Datafeed Toolbox uses the following set of core functions to manage communication:

- `bloomberg`, `datastream`, `factset`, `hyperfeed`, `idc`, or `yahoo`: Establishes a connection to the appropriate data server.
- `isconnection`: Verifies that a connection is working.
- `get`: Retrieves connection properties.
- `close`: Terminates the connection.

An additional function, `fetch`, obtains the desired data from the data server and transfers it to your PC.

Example: The bloomberg Function

Connect to the Bloomberg data server using the `bloomberg` function. The connection requires a port number and an IP address.

The syntax for the `bloomberg` function is

```
Connect = bloomberg(PortNumber, 'IPAddress')
```

The IP address is entered as a MATLAB string. For example, the expression

```
c = bloomberg(8194, '123.456.54.123')
```

returns a Bloomberg connection object:

```
c =  
  
    connection: 84554360  
    ipaddress: '123.456.54.123'  
    port: 8194
```

The `connection` field within the object `c` contains the Bloomberg connection handle that will be used in processing future data requests.

If you want to accept the default port number and IP address provided when your Bloomberg software was installed, enter

```
c = bloomberg
```

with no arguments.

Verifying the Connection

To verify that a data server connection is valid and open, use the `isconnection` function. For a connection object `c` previously created with one of the above connection functions,

```
x = isconnection(c)
```

returns `x = 1` if the connection is valid and open or `x = 0` if the connection is closed or invalid.

Retrieving Connection Properties

To retrieve the properties of a connection object, use the function `get`. This function returns different values depending upon which data server is being used.

Example: Retrieving Bloomberg Connection Properties

For the Bloomberg connection

```
c = bloomberg(8194, '123.456.54.123')
```

the command

```
p = get(c)
```

returns the list of all valid connection properties and their values associated with the connection object `c`:

```
p =  
  connection: 84554360  
  ipaddress: '123.456.54.123'  
  port: 8194  
  socket: 248  
  version: 1.8000
```

The `get` function can return specific properties of a connection object. For example, to obtain the port number and Bloomberg version for the connection object `c`, use the format

```
p = get(c,{'Port';'Version'})
```

which returns

```
p =  
    port: 8194  
    version: 1.8000
```

When returning a single property, for example, the connection handle, the function

```
p = get(c,'Connection')
```

returns

```
p =  
84554360
```

For a single returned property the output is not a structure.

Disconnecting from a Data Server

To close a data server connection and disconnect, use the `close` function with the format

```
close(Connect)
```

You must have previously created the connection object with one of the connection functions.

Retrieving Data

The `fetch` function controls data retrieval from a data server connection. `fetch` returns different information depending upon which data server is being accessed. See the version of `fetch` appropriate for your data server for further information.

Example: Retrieving Bloomberg Data

This section illustrates the use of the `fetch` function to retrieve data from a Bloomberg data server. Versions of the `fetch` function that retrieve data from other data servers work similarly.

Retrieving Header (Bloomberg Default) Data

A header (default) data request to Bloomberg returns a fixed set of field data. Not all fields in the header data are relevant for a specific security.

Determining Header Fields. The list of valid header fields is stored in the file `@bloomberg/bbfields.mat`. Use the MATLAB load command

```
load @bloomberg/bbfields
```

to load this file. The variable `headerfieldnames` contains the list of header field names.

Obtaining Data. To retrieve header data from the Bloomberg connection, use `fetch` with the syntax

```
data = fetch(Connect, 'Security', 'HEADER', 'Flag')
```

where

- `Connect` is a Bloomberg connection object established with the `bloomberg` function.
- `Security` is the list of securities for which data is requested.

Note Security names are case sensitive for Bloomberg `fetch`.

- The 'HEADER' argument is entered literally.
- Flag denotes the dates for which data can be retrieved. Flag has three possible values:
 - DEFAULT fills all fields with data from the most recent date with a bid, ask, or trade.
 - TODAY fills the fields with data from today only.
 - ENHANCED fills the fields with data for the most recent event for each individual field. In this case, for example, the bid and ask group fields could come from different dates.

Commands of the form

```
data = fetch(Connection, Security)
data = fetch(Connection, Security, 'HEADER')
data = fetch(Connection, Security, 'HEADER', 'DEFAULT')
```

are equivalent.

The returned data has a fixed set of fields. For example, a header inquiry for the security IBM US Equity returns data of the form:

```
Status:0
      Open:93
TodaysOpenPrice:93
      HighPrice:93.1875
TodaysHighPrice:93.1875
      LowPrice:89
TodaysLowPrice:89
      LastPrice:90.9375
TodaysLastPrice:0
      SettlePrice:NaN
      BidPrice:0
TodaysBidPrice:NaN
      AskPrice:0
TodaysAskPrice:NaN
      YieldBid:NaN
TodaysYieldBid:NaN
      YieldAsk:NaN
```

```
TodaysYieldAsk:NaN
  LimitUp:NaN
  LimitDown:NaN
  OpenInterest:3359000
LastPriceYesterday:95
  Scale:1
  LastPriceTime:0.4993
LastTradeExchange:7
  TickDirection:-1
  BidSize:0
  TodaysBidSize:NaN
  AskSize:NaN
  TodaysAskSize:0
  BidCondition:NaN
  AskCondition:NaN
  LastTradeCondition:NaN
  LastMarketCondition:NaN
  Monitorable:1
  TotalVolume:60018500
  TodaysTotalVolume:0
  TotalNumberOfTicks:63318
  TodaysTotalNumberOfTicks:63318
  SessionStartTime:0.3958
  SessionEndTime:0.6875
  Currency:538989397
  Format:0
  SecurityKey: {'IBM US Equity'}
  AsOfDate:730441
  TodaysAsOfDate:730441
```

Not all fields are applicable to IBM US Equity, the security about which we inquired.

Retrieving Field Data

The `fetch` function with the `GETDATA` argument obtains Bloomberg field data. The entire set of field data provides statistics for all possible securities but does not apply universally to any one security.

Determining Field Names. The file @bloomberg/bbfields.mat stores the complete list of valid field names. Use the function

```
load @bloomberg/bbfields
```

to load this file. You will see a list of four variables:

```
bbcategories  
bbfieldids  
bbfieldnames  
headerfieldnames
```

The variable bbfieldnames contains a list of field names. This list includes the header field names plus numerous others. The other variables loaded extend the list of field names.

Obtaining Data. To obtain data for specific fields of a given security, use the fetch function with the syntax

```
d = fetch(Connect, Security, 'GETDATA', Fields)
```

For example, use the bloomberg function to establish a connection c1 to a Bloomberg data server.

```
c1 = bloomberg(8234, '123.457.78.999')
```

Then

```
d = fetch(c1, 'IBM US Equity', 'GETDATA', {'Open'; 'Last_Price'})
```

returns

```
d =  
      Open: 126.2500  
      Last_Price: 125.1250
```


Retrieving Time Series Data

The `fetch` function with the `TIMESERIES` argument returns price and volume data for a particular security on a specified date. Time series data for a given security and a specific date are returned using the syntax

```
data = fetch(Connection, Security, 'TIMESERIES', Date)
```

Date may be a MATLAB date string or serial date number.

To obtain time series data for the current day, you can use the alternate form of the function

```
data = fetch(Connection, Security, 'TIMESERIES', now)
```

To obtain time series data for IBM using an existing connection `c1`, enter the function

```
data = fetch(c1, 'IBM US Equity', 'TIMESERIES', '11/16/99')
```

The result will look like this:

```
data =
    31.00    730440.31    130.00    1000.00
    32.00    730440.31    130.00     200.00
    32.00    730440.35    129.50   10000.00
    31.00    730440.35    129.50     100.00
    32.00    730440.35    129.50     100.00
     1.00    730440.56    129.25    4000.00
    31.00    730440.56    129.38    1500.00
    32.00    730440.56    129.50     500.00
     1.00    730440.56    129.63    5000.00
    31.00    730440.56    129.63     400.00
    32.00    730440.56    129.63     200.00
     1.00    730440.56    129.69    5000.00
    31.00    730440.56    129.69     500.00
    32.00    730440.56    129.69     500.00
    31.00    730440.56    129.75     100.00
    32.00    730440.56    130.00     100.00
     1.00    730440.56    130.00    5000.00
     1.00    730440.56    129.88    5000.00
```

```
31.00      730440.56      129.88      300.00
```

Column 1 contains the tick type flag, column 2 contains the time stamp in MATLAB serial date number format, column 3 contains the tick value, and column 4 contains the number of shares in the transaction.

Retrieving Historical Data

Use the `fetch` function with the `HISTORY` argument to obtain historical data for a specific security.

For a specified field of a particular security use the syntax

```
d = fetch(Connect,Security,'HISTORY',Field,FromDate,ToDate)
```

to obtain historical data. Data for the field is returned for the date range from `FromDate` to `ToDate`. See “Determining Field Names” on page 1-10 for instructions on determining valid field names.

For example, to obtain the closing price for IBM for the dates July 15, 1999 to August 2, 1999 using the connection `c1`, enter

```
data = fetch(c1, 'IBM US Equity', 'HISTORY', 'Last_Price',...  
'07/15/99', '08/02/99')
```

```
data =
```

```
730316.00      136.31  
730317.00      136.25  
730320.00      134.63  
730321.00      128.25  
730322.00      129.00  
730323.00      123.88  
730324.00      124.81  
730327.00      123.00  
730328.00      126.25  
730329.00      128.38  
730330.00      125.38  
730331.00      125.69  
730334.00      122.25
```

Column 1 contains the date represented as a MATLAB date number, and column 2 contains the last price.

Finding Ticker Symbols

You can use the `fetch` function with the `LOOKUP` argument to find a ticker symbol when you are uncertain what the symbol might be. Use the syntax

```
data = fetch(Connect, SearchString, 'LOOKUP', Market)
```

to locate a specific ticker symbol.

The `SearchString` argument is the comparison string used in the lookup operation, and `Market` indicates the type of security (the market in which the security trades). The allowable values for `Market` are

- `Comdty` (Commodities)
- `Corp` (Corporate bonds)
- `Curncy` (Currencies)
- `Equity` (Equities)
- `Govt` (Government bonds)
- `Index` (Indexes)
- `M-Mkt` (Money Market securities)
- `Mtge` (Mortgage-backed securities)
- `Muni` (Municipal bonds)
- `Pfd` (Preferred stocks)

For example, using `fetch` with the connection `c1` to look up the ticker symbol for New Zealand government bonds

```
data = fetch(c1, 'New', 'LOOKUP', 'Govt')
```

returns a list of possible values:

```
data =  
  
'NZTB   New Zealand Treasury Bill NZGB   New Zealand Governme'  
'NZGB   New Zealand Government Bond NZ   New Zealand Govern'  
'NZ     New Zealand Government International Bond HCNZ   Hous'  
'ECNZ   Electric Corporation of New Zealand Bond NZTB NZGB NZ H'
```

Datafeed Toolbox Graphical User Interface

The Datafeed Toolbox provides a graphical user interface (GUI) consisting of two dialog boxes. The Datafeed dialog box consists of two tabbed dialog boxes, one to establish a data server connection, and the second to retrieve data from the server. The second dialog box, the Securities Lookup dialog box, lets you find the ticker symbol for a specific security when you know at least part of the name of the security.

For additional information about the Datafeed dialog box, see

- “Connecting to a Data Server” on page 1-15
- “Data Retrieval” on page 1-18

To learn about setting overrides on retrieved data, see

- “Setting Overrides” on page 1-20

For additional information about the Securities Lookup dialog box, see

- “Securities Lookup Dialog Box (Bloomberg, FT Interactive Data)” on page 1-21

Datafeed Dialog Box

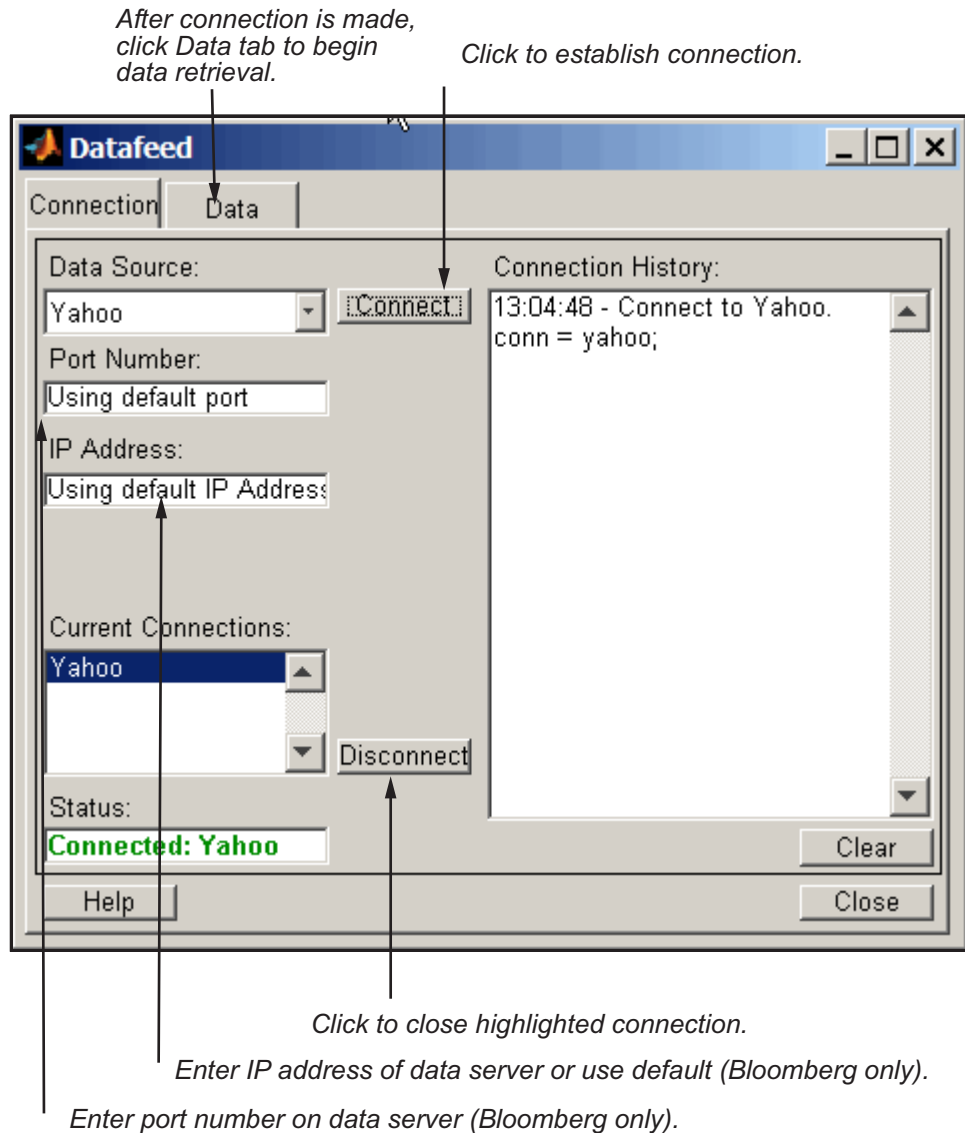
The Datafeed dialog box establishes the connection with the data server and manages the retrieval of data. Enter the function `df tool` to display the Datafeed dialog box on your screen. The Datafeed dialog box consists of two tabbed dialog boxes:

- The **Connection** tab establishes communication with a data server. (See “Connecting to a Data Server” on page 1-15.)
- The **Data** tab specifies the data request. (See “Data Retrieval” on page 1-18.)

Connecting to a Data Server

The **Connection** tab establishes a connection to one or more data servers. For FactSet, Federal Reserve Economic Data, Yahoo!, and FT Interactive Data

connections, choose the data server from the **Data Source** choices and click the **Connect** button. For a Bloomberg connection, you can specify a specific IP address and port number on the Bloomberg server, or alternatively, just click the **Connect** button and accept the default values provided when the Bloomberg software was installed on your machine.



- 1** (Bloomberg only) Enter the port number on the data server in the **Port Number** box (or use default).
- 2** (Bloomberg only) Enter the IP address of the data server in the **IP Address** box (or use default).
- 3** Click the **Connect** button to establish the connection.
- 4** When the Connected message appears in the **Status** box, click the **Data** tab to begin the process of retrieving data from the data server. (For information on the **Data** tab, see “Data Retrieval” on page 1-18.)
- 5** Click the **Disconnect** button to terminate the session highlighted in the **Current Connections** box.

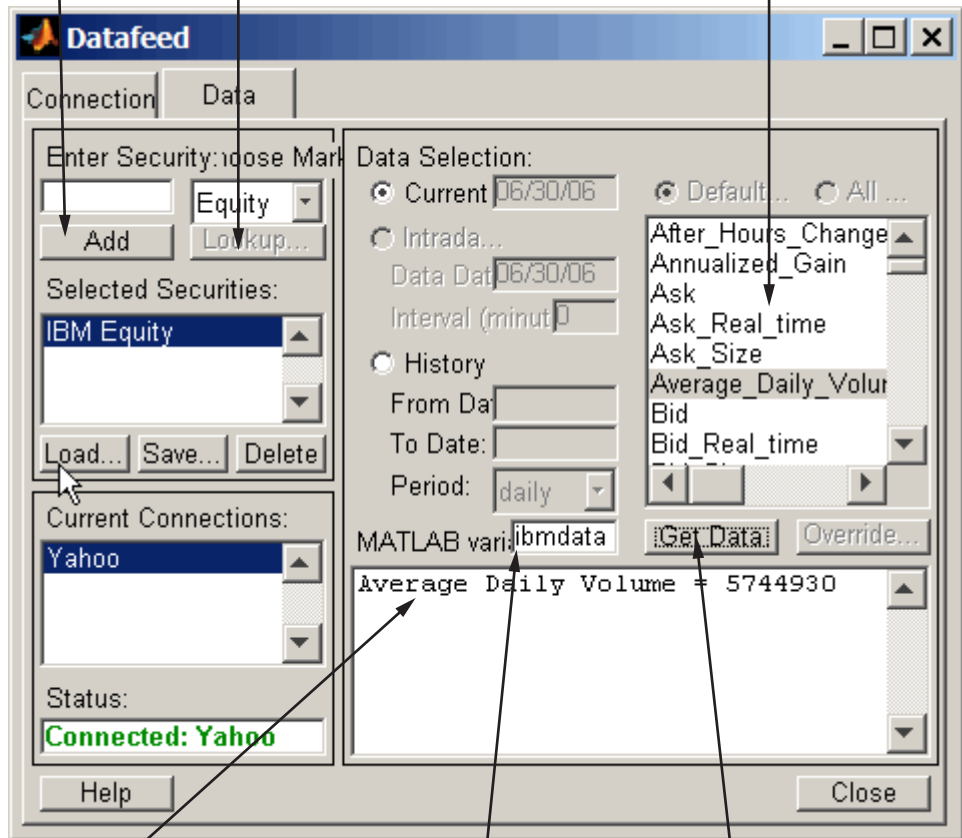
Data Retrieval

The **Data** tab manages the retrieval of data from the data server. It also allows you to access a dialog box to set overrides on the data.

Enter security symbol if known.
 Click Get Data button to retrieve data.
 Click Add button to add security to Selected Securities list.

(Bloomberg only)
 Use to find security symbol if known.

Security fields.



Fields with data retrieved from the connection.

Variable in MATLAB workspace.

Click to retrieve data.

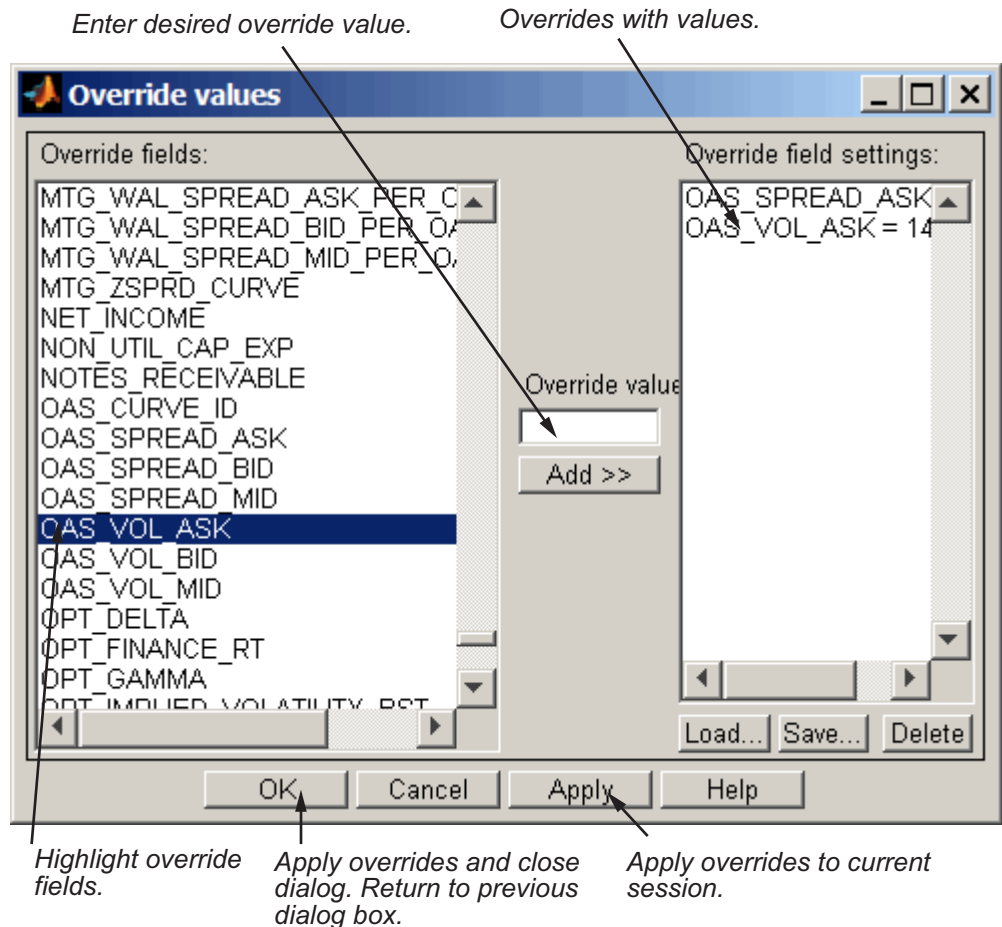
1 Enter the security symbol in the **Enter Security** box.

- 2** Indicate the type of data you are seeking in the **Data Selection** pane.
- 3** Indicate whether you want the default or full set of data in the **Fields** pane.
- 4** Click the **Get Data** button to retrieve data from the data server.
- 5** Click the **Override** button if you want to set overrides on the data you request from the data server.

Note If you do not know the symbol for a security, you can use the **Lookup** button to find the name of the security. (See “Securities Lookup Dialog Box (Bloomberg, FT Interactive Data)” on page 1-21.)

Setting Overrides

Click the **Override** button if you want to set overrides on the data you obtain. The Override values dialog box will open.



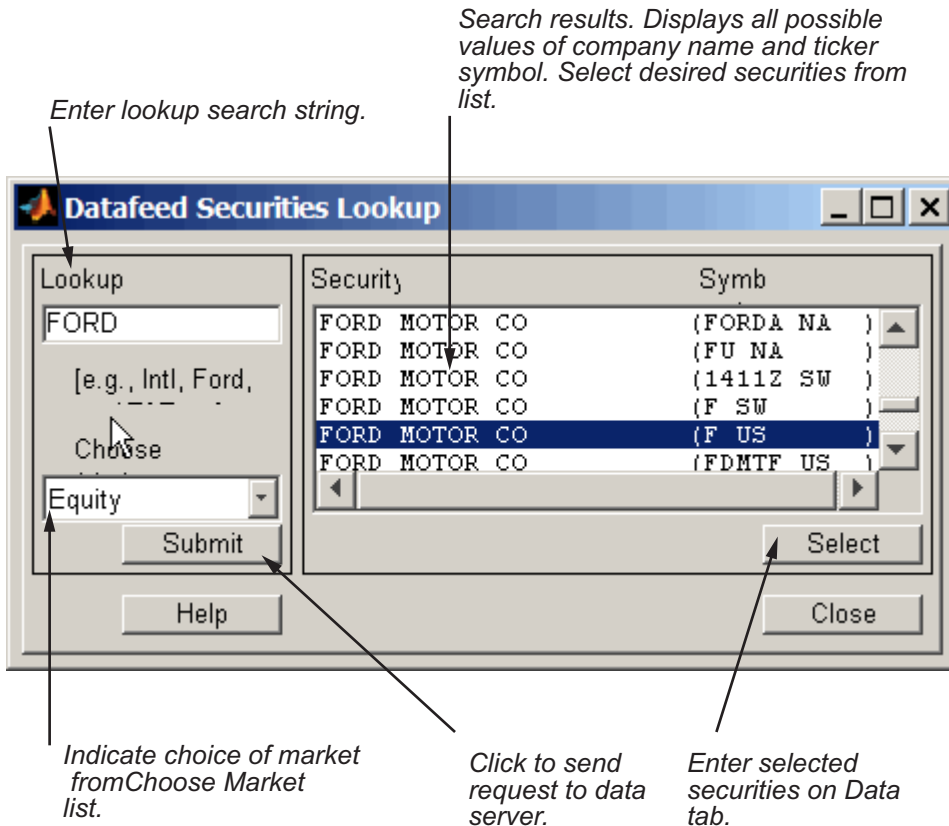
Securities Lookup Dialog Box (Bloomberg, FT Interactive Data)

Click the **Lookup** button of the Datafeed dialog box **Data** tab to display the Securities Lookup dialog box. See “Data Retrieval” on page 1-18 for information about the **Data** tab.

The Securities Lookup dialog box provides a means to obtain the ticker symbol for a particular security when you know part of the name. You can then

enter the ticker symbol into the **Enter Security** field on the **Data** tab. It is essential that you enter the ticker symbol as specified; otherwise, the data server may provide no data or provide data for some other security.

Alternatively, you can highlight one or more securities in the list and click **Select**. The selected securities are added to the **Selected Securities** list on the **Data** tab.



Functions — By Category

Bloomberg (p. 3-2)

Get Bloomberg financial data

Thomson Datastream (p. 3-22)

Get Thomson Datastream financial data

FactSet (p. 3-30)

Get FactSet financial data

Hyperfeed (p. 3-39)

Get Hyperfeed financial data

FT Interactive Data (p. 3-48)

Get FT Interactive Data financial data

Federal Reserve Economic Data
(p. 3-55)

Get Federal Reserve Economic Data
financial data

Yahoo! (p. 3-63)

Get Yahoo! financial data

Bloomberg

bloomberg	Connect to Bloomberg
bloomberg	Connect to Bloomberg
close	Close Bloomberg connection
close	Close Bloomberg connection
fetch	Request data from Bloomberg
fetch	Request data from Bloomberg
get	Bloomberg connection properties
get	Bloomberg connection properties
isconnection	True if valid Bloomberg connection
isconnection	True if valid Bloomberg connection
pricevol	Price and volume (demonstration)
pricevol	Price and volume (demonstration)
showtrades	Recent trade data (demonstration)
showtrades	Recent trade data (demonstration)
stockticker	Trades with volumes (demonstration)
stockticker	Trades with volumes (demonstration)

Purpose Connect to Bloomberg

Syntax `Connect = bloomberg(PortNumber, 'IPAddress')`
`Connect = bloomberg`

Arguments

PortNumber	Port on machine where connection is being made.
IPAddress	A MATLAB string containing the Internet address of machine where connection is being made.

Description `Connect = bloomberg(PortNumber, 'IPAddress')` establishes a connection to a Bloomberg data server using the port number, PortNumber, and the Internet address, IPAddress.

`Connect = bloomberg` establishes a connection to a Bloomberg data server using port number 8194 and the default Internet address provided when the Bloomberg software was installed on your machine.

Examples `c = bloomberg(8194, '111.222.33.444')`

makes a connection to the Bloomberg server on port 8194 of the machine with Internet address 111.222.33.444.

See Also `close`, `fetch`, `get`, `isconnection` (Bloomberg functions)

close

Purpose	Close Bloomberg connection		
Syntax	<code>close(Connect)</code>		
Arguments	<table><tr><td><code>Connect</code></td><td>Bloomberg connection object created with the bloomberg function.</td></tr></table>	<code>Connect</code>	Bloomberg connection object created with the bloomberg function.
<code>Connect</code>	Bloomberg connection object created with the bloomberg function.		
Description	<code>close(Connect)</code> closes the connection to the Bloomberg data server.		
Examples	<pre>c = bloomberg(8194, '111.222.33.444')</pre> <p>establishes a Bloomberg connection, <code>c</code>.</p> <pre>close(c)</pre> <p>closes this connection.</p>		
See Also	<code>bloomberg</code>		

Purpose

Request data from Bloomberg

Syntax

```
data = fetch(Connect, 'Security')
data = fetch(Connect, 'Security', 'HEADER', 'Flag', 'Ident')
data = fetch(Connect, 'Security', 'GETDATA', 'Fields',
'Override', 'Ident', 'Values')
data = fetch(Connect, 'Security', 'TIMESERIES', 'Date',
'Minutes', 'TickField')
data = fetch(Connect, 'Security', 'HISTORY', 'Fields',
'FromDate', 'ToDate', 'Period', 'Currency', 'Ident')
ticker = fetch(Connect, 'SearchString', 'LOOKUP', 'Market')
data = fetch(Connect, 'Security', 'REALTIME', 'Fields',
MATLABProg)
data = fetch(Connect, Security, 'STOP')
```

Arguments

Connect	Bloomberg connection object created with the bloomberg function.
Security	A MATLAB string containing the name of a security in a format recognizable by the Bloomberg server. You can substitute a CUSIP number for a security name if you want.

Note The Security argument may be a cell array of strings containing a list of securities.

<i>Flag</i>	A MATLAB string indicating the dates from which data is to be retrieved. Possible values are DEFAULT: Data from most recent bid, ask, or trade. If a <i>Flag</i> value is not specified, 'DEFAULT' is assumed. TODAY: Today's data only. ENHANCED: Data from most recent date of each individual field.
<i>Currency</i>	(Optional) Currency in which historical returns are provided. Valid currencies are listed in the file @bloomberg/bbfields.mat. Default = [].
<i>Ident</i>	(Optional) Security type identifier. Valid security type identifiers are listed in the file @bloomberg/bbfields.mat. Default = [].
<i>Fields</i>	A MATLAB string or cell array of strings indicating specific fields for which data is requested. Valid field names are listed in the file @bloomberg/bbfields.mat. The variable bbfieldnames contains the list of field names. Default = [].
<i>Override</i>	(Optional) String or cell array of strings containing override field list. Default = [].
<i>Values</i>	(Optional) String or cell array of strings containing override field values.
<i>Date</i>	Date string or serial date number indicating date for the time series. Specify now for today's time series data.
<i>Minutes</i>	(Optional) Tick interval in minutes.
<i>TickField</i>	(Optional) The field can be specified as a string or numeric value (e.g., TickField = 'Trade' or TickField = 1 return data for ticks of type Trade. Use the function dftool('ticktypes') to return the list of intraday tick fields.

`FromDate` Beginning date for historical data.

Note Dates can be specified in any of the formats supported by `datestr` and `datenum` that show a year, month and day.

`ToDate` End date for historical data.

Period (Optional) Period of the data. A MATLAB three-part string with the format:

'Frequency Days Data'

Frequency Values

d: daily (default)
w: weekly
m: monthly
q: quarterly
y: yearly

Days Values

o: omit all days for which there is no data (default)
i: include all trading days
a: include all calendar days

Data Values

b: report missing data using Bloomberg (default)
s: show missing data as last found value
n: report missing data as NaN

For example, 'dan' returns daily data for all calendar days, reporting missing values as NaN. If a value is unspecified (e.g., ' n'), the unspecified values are replaced by defaults.

Note If *Period* is not specified, default values are used.

<i>Currency</i>	(Optional) Currency type. The file @bloomberg/bbfields.mat lists the supported currencies.
<i>Market</i>	A MATLAB string indicating the market in which a particular security trades. <i>Market</i> values are Comdty: (Commodities) Corp: (Corporate bonds) Equity: (Equities) Govt: (Government bonds) Index: (Indexes) M-Mkt: (Money Market securities) Mtge: Mortgage-backed securities) Muni: (Municipal bonds) Pfd: (Preferred stocks)
MATLABProg	A string that is the name of any valid MATLAB program.

Description

For a given security, `fetch` returns header (default), current, time series, real time, and historical data via the Bloomberg connection.

`data = fetch(Connect, 'Security')` fills the header fields with data from the most recent date with a bid, ask, or trade.

`data = fetch(Connect, 'Security', 'HEADER', 'Flag', 'Ident')` returns data for the most recent date of each individual field for the specified security type identifiers, based upon the value of *Flag*.

- If *Flag* is DEFAULT, fetch fills the header fields with data from the most recent date with a bid, ask, or trade. This is the equivalent of `data = fetch(Connect, 'Security')`.
- If *Flag* is TODAY, fetch returns the header field data with data from today only.
- If *Flag* is ENHANCED, fetch returns the header field data for the most recent date of each individual field. In this case, for example, the bid and ask group fields could come from different dates.

`data = fetch(Connect, 'Security', 'GETDATA', 'Fields', 'Override', 'Ident', 'Values')` returns the current market data for the specified fields of the indicated security. You can further specify the data with the optional *Override*, *Values* and *Ident* arguments.

`data = fetch(Connect, 'Security', 'TIMESERIES', 'Date', 'Minutes', 'TickField')` returns the tick data for a security for the specified date. You can further specify the data with the optional *Minutes* and *TickField* arguments.

You can specify *TickField* as a string or numeric value, e.g., `TickField = 'Trade'` or `TickField = 1` returns data for ticks of type Trade. The function `dftool('ticktypes')` returns the list of intraday tick fields. Intraday tick data requested with an interval is returned with the columns representing

- Time
- Open
- High
- Low
- Value of last tick
- Volume total value of ticks
- Total value of ticks for the time range
- Number of ticks

Columns 7 and 8 are provided only if they make sense for the requested field.

For today's tick data, specify

```
data = fetch(Connect, 'Security', 'TIMESERIES', now)
```

For today's trade time series aggregated into five-minute intervals, enter

```
data = fetch(Connect, 'Security', 'TIMESERIES', ...  
now, 5, 'Trade')
```

```
data = fetch(Connect, 'Security', 'HISTORY', 'Fields',  
'FromDate', 'ToDate', 'Period', 'Currency', 'Ident')
```

 returns historical data for the specified field for the date range FromDate to ToDate. You can further specify the date range by setting the time period with the optional *Period* argument. You can further specify the data to be returned by appending the *Currency* or *Ident* argument.

```
ticker = fetch(Connect, 'SearchString', 'LOOKUP', 'Market')
```

 uses SearchString to find the ticker symbol for a security trading in a designated market. The output ticker is a column vector of possible ticker values.

Note If you supply *Ident* without a period or currency, enter [] for the missing values.

```
data = fetch(Connect, 'Security', 'REALTIME', 'Fields',  
MATLABProg)
```

 subscribes to a given security or list of securities, requesting the indicated fields, and runs any specified MATLAB function. See pricevol, showtrades, or stockticker for information on the data returned by asynchronous Bloomberg events.

```
data = fetch(Connect, Security, 'STOP')
```

 unsubscribes the list of securities from processing Bloomberg real-time events.

Examples

Returning Header Data

```
D = fetch(C, 'ABC US Equity')
```

returns the header data for a United States equity with ticker ABC.

Opening and Closing Prices

```
D = fetch(C, 'ABC US Equity', 'GETDATA', ...  
        {'Last_Price'; 'Open'})
```

returns the opening and closing prices.

Override Fields

```
D = fetch(C, '3358ABCD4 Corp', 'GETDATA', ...  
        {'YLD_YTM_ASK', 'ASK', 'OAS_SPREAD_ASK', 'OAS_VOL_ASK'}, ...  
        {'PX_ASK', 'OAS_VOL_ASK'}, {'99.125000', '14.000000'})
```

returns the requested fields given override fields and values.

Time Series

```
D = fetch(C, 'ABC US Equity', 'TIMESERIES', now)
```

return today's time series.

Time Intervals

```
D = fetch(C, 'ABC US Equity', 'TIMESERIES', now, 5, 'Trade')
```

returns today's trade time series for the given security aggregated into five-minute intervals.

Default Closing Price

```
D = fetch(C, 'ABC US Equity', 'HISTORY', 'Last_Price', ...  
        '8/01/99', '8/10/99')
```

returns the closing price for the given dates using the default period of the data.

fetch

Monthly Closing Price

```
D = fetch(C, 'ABC US Equity', 'HISTORY', 'Last_Price', ...  
         '8/01/99', '9/30/00', 'm')
```

returns the monthly closing price for the given dates for the given security.

See Also

bloomberg, close, get, isconnection (Bloomberg functions)

Purpose Bloomberg connection properties

Syntax
`value = get(Connect, 'PropertyName')`
`value = get(Connect)`

Arguments

<code>Connect</code>	Bloomberg connection object created with the bloomberg function.
<code>PropertyName</code>	(Optional) A MATLAB string or cell array of strings containing property names. Property names are Connection IPAddress Port Socket Version

Description `value = get(Connect, 'PropertyName')` returns a MATLAB structure containing the value of the specified properties for the Bloomberg connection object.

`value = get(Connect)` returns the value for all properties.

Examples

```
c = bloomberg(8194, '111.222.33.444')
```

establishes a Bloomberg connection, `c`.

The syntax

```
p = get(c, {'Port', 'IPAddress'})
```

get

returns

```
p =  
  port: 8194  
  ipaddress: 111.222.33.444
```

See Also

bloomberg, close, fetch, isconnection (Bloomberg functions)

Purpose	True if valid Bloomberg connection		
Syntax	<code>x = isconnection(Connect)</code>		
Arguments	<table><tr><td>Connect</td><td>Bloomberg connection object created with the bloomberg function.</td></tr></table>	Connect	Bloomberg connection object created with the bloomberg function.
Connect	Bloomberg connection object created with the bloomberg function.		
Description	<code>x = isconnection(Connect)</code> returns <code>x = 1</code> if the connection is a valid Bloomberg connection, and <code>x = 0</code> if it is not.		
Examples	<p>The function</p> <pre>c = bloomberg(8194, '111.222.33.444')</pre> <p>establishes a Bloomberg connection, <code>c</code>.</p> <p>Then</p> <pre>x = isconnection(c) x = 1</pre> <p>indicates that <code>c</code> is a valid Bloomberg connection.</p>		
See Also	<code>bloomberg</code> , <code>close</code> , <code>fetch</code> , <code>get</code> (Bloomberg functions)		

pricevol

Purpose Price and volume (demonstration)

Syntax pricevol(InputList)

Arguments

InputList Fields for which real-time data is sought.

Description

pricevol(InputList) demonstrates the Bloomberg real-time data import functionality. InputList is an input list of the elements:

InputList(1) = COM.Bloomberg.Data.1	Bloomberg handle
InputList(2) = 1	Event ID
InputList(3) = ('Security')	Security string
InputList(4) = 1	Cookie
InputList(5) = 2	Field number ID
InputList(6) = {[43.58]}	Return data for the given tick
InputList(7) = 0	Status
InputList(8)	Structure containing the above fields
InputList(9) = 'Data'	Event type

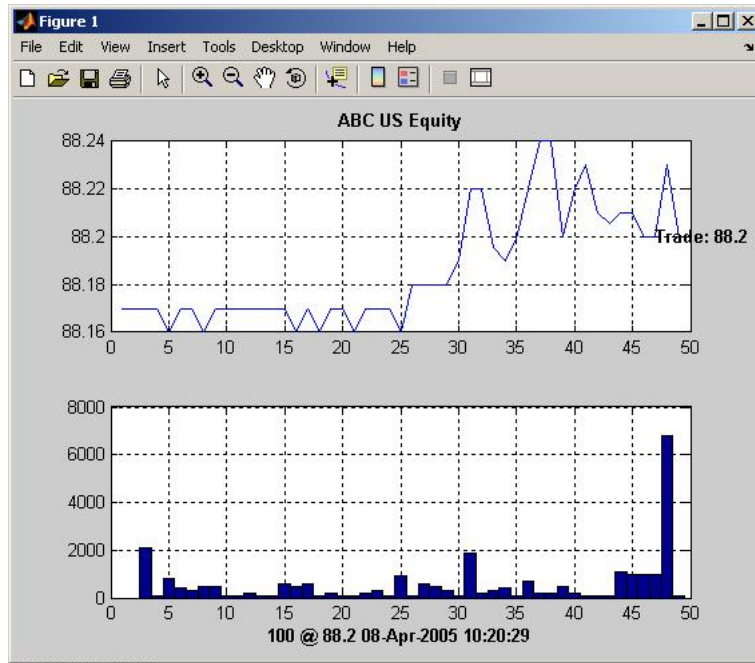
The input argument InputList(8) contains the necessary information to process real-time events.

Examples

The following example shows the use of this function.

```
b = bloomberg;  
d = fetch(b, 'ABC US Equity', 'REALTIME', ...  
{ 'Last_Trade', 'Volume' }, 'pricevol');
```

The output displays the most recent Trade and Volume in the figure and shows the most recent trade with volumes.



See Also `showtrades`, `stockticker`

showtrades

Purpose Recent trade data (demonstration)

Syntax showtrades(InputList)

Arguments

InputList Fields for which real-time data is sought.

Description

showtrades(InputList) demonstrates the Bloomberg real-time data import functionality. InputList is an input list of the elements:

InputList(1) = COM.Bloomberg.Data.1	Bloomberg handle
InputList(2) = 1	Event ID
InputList(3) = ('Security')	Security string
InputList(4) = 1	Cookie
InputList(5) = 2	Field number ID
InputList(6) = {[43.58]}	Return data for the given tick
InputList(7) = 0	Status
InputList(8)	Structure containing the above fields
InputList(9) = 'Data'	Event type

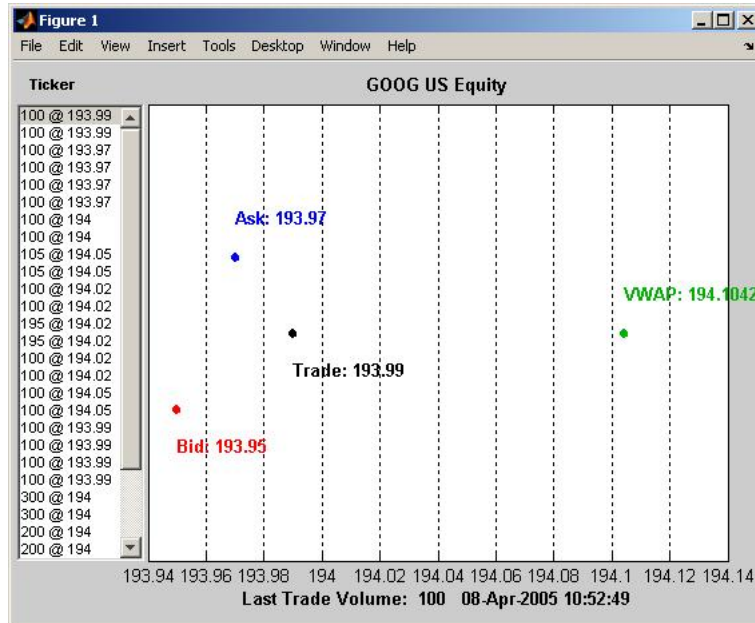
The input argument InputList(8) contains the necessary information to process real-time events.

Examples

The following example shows the use of this function.

```
b = bloomberg;  
d = fetch(b, 'GOOG US Equity', 'REALTIME', ...  
{ 'Last_Trade', 'Bid', 'Ask', 'Volume', 'VWAP' }, 'showtrades');
```

The output shows the most recent Trade, Bid, Ask and VWAP (volume-weighted adjusted price) and a list of the most recent trades with volumes.



See Also pricevol, stockticker

stockticker

Purpose Trades with volumes (demonstration)

Syntax stockticker(InputList)

Arguments

InputList Fields for which real-time data is sought.

Description

stockticker(InputList) demonstrates the Bloomberg real-time data import functionality. InputList is an input list of the elements:

InputList(1) = COM.Bloomberg.Data.1	Bloomberg handle
InputList(2) = 1	Event ID
InputList(3) = ('Security')	Security string
InputList(4) = 1	Cookie
InputList(5) = 2	Field number ID
InputList(6) = {[43.58]}	Return data for the given tick
InputList(7) = 0	Status
InputList(8)	Structure containing the above fields
InputList(9) = 'Data'	Event type

The input argument InputList(8) contains the necessary information to process real-time events.

Examples

The following example shows the use of this function. The output provides a list of trades with volumes for each requested security.

```
b = bloomberg;  
d = fetch(b,{'IBM US Equity','EMC US Equity','NTAP US Equity'},...  
'REALTIME',{'Last_Trade','Volume'},'stockticker');  
** EMC US Equity ** 0 @ 12.65 08-Apr-2005 10:24:57
```



```
** IBM US Equity ** 0 @ 88.17 08-Apr-2005 10:24:57
** NTAP US Equity ** 0 @ 29.02 08-Apr-2005 10:24:57
** EMC US Equity ** 200 @ 12.66 08-Apr-2005 10:24:58
** EMC US Equity ** 1400 @ 12.65 08-Apr-2005 10:24:58
** EMC US Equity ** 3100 @ 12.66 08-Apr-2005 10:25:00
** IBM US Equity ** 1300 @ 88.17 08-Apr-2005 10:25:00
.
.
.
```

See Also pricevol, showtrades

Thomson Datastream

close	Close Thomson Datastream connection
close	Close Thomson Datastream connection
datastream	Thomson Datastream API connection
datastream	Thomson Datastream API connection
fetch	Request data from Thomson Datastream
fetch	Request data from Thomson Datastream
get	Thomson Datastream connection properties
get	Thomson Datastream connection properties
isconnection	True if valid Thomson Datastream connection
isconnection	True if valid Thomson Datastream connection

Purpose	Close Thomson Datastream connection		
Syntax	<code>close(Connect)</code>		
Arguments	<table><tr><td>Connect</td><td>Thomson Datastream connection object created with the <code>datastream</code> function.</td></tr></table>	Connect	Thomson Datastream connection object created with the <code>datastream</code> function.
Connect	Thomson Datastream connection object created with the <code>datastream</code> function.		
Description	<code>close(Connect)</code> closes the connection to the Thomson Datastream data server.		
See Also	<code>datastream</code>		

datastream

Purpose	Thomson Datastream API connection	
Syntax	Connect = datastream('UserName', 'Password', 'Source', 'URL')	
Arguments	UserName	User name.
	Password	User password.
	Source	To connect to the Thomson Datastream API, enter 'Datastream' in this field.
	URL	Web URL.

Note Thomson assigns the values you need to enter for each argument. Enter all arguments as MATLAB strings.

Description Connect = datastream('UserName', 'Password', 'Source', 'URL') makes a connection to the Thomson Datastream API, which provides access to the Thomson Datastream content.

Examples Use the datastream function to establish a connection to the Thomson Datastream API, which provides you access to the Thomson Datastream content.

```
Connect = datastream('User1', 'Pass1', 'Datastream', ...  
    'http://dataworks.thomson.com/Dataworks/Enterprise/1.0')
```

See Also close, fetch, get, isconnection (Thomson Datastream functions)

Purpose

Request data from Thomson Datastream

Syntax

```
data = fetch(Connect, 'Security')
data = fetch(Connect, 'Security', 'Fields')
data = fetch(Connect, 'Security', 'Fields', 'Date')
data = fetch(Connect, Security, 'Fields', 'FromDate',
            'ToDate')
data = fetch(Connect, Security, 'Fields', 'FromDate',
            'ToDate', 'Period')
data = fetch(Connect, Security, 'Fields', 'FromDate',
            'ToDate', 'Period', 'Currency')
```

Arguments

Connect	Thomson Datastream connection object created with the <code>datastream</code> function.
Security	MATLAB string containing the name of a security in a format recognizable by the Thomson Datastream data server.
<i>Fields</i>	(Optional) MATLAB string or cell array of strings indicating the data fields for which data is to be retrieved.
Date	(Optional) MATLAB string indicating a specific calendar date for which data is requested.

fetch

FromDate (Optional) Start date for historical data.

Note Dates can be specified in any of the formats supported by `datestr` and `datenum` that show a year, month and day.

ToDate (Optional) End date for historical data. If *ToDate* is provided, *FromDate* cannot be an empty value.

Period (Optional) Period within a date range. *Period* values are
'd': daily values
'w': weekly values
'm': monthly values

Currency (Optional) Currency in which the data is reported.

Note You can input the optional arguments *Fields*, *FromDate*, *ToDate*, *Period*, and *Currency* as MATLAB strings or empty arrays ([]).

Description

`data = fetch(Connect, 'Security')` returns the default time series for the indicated security.

`data = fetch(Connect, 'Security', 'Fields')` returns data for the specified security and fields.

`data = fetch(Connect, 'Security', 'Fields', 'Date')` returns data for the specified security and fields on a particular date.

```
data = fetch(Connect, Security, 'Fields', 'FromDate',  
'ToDate') returns data for the specified security and fields for the  
indicated date range.
```

```
data = fetch(Connect, Security, 'Fields', 'FromDate',  
'ToDate', 'Period') returns instrument data for the given range  
with the indicated period.
```

```
data = fetch(Connect, Security, 'Fields', 'FromDate',  
'ToDate', 'Period', 'Currency') additionally specifies the currency  
in which the data is reported.
```

Examples

Here are some examples of using the Datastream fetch command to obtain data from Thomson Datastream.

Note The Datastream interface returns all data as strings. For example, when requesting Price data, it is returned in MATLAB as a cell array of strings within the structure. There is no way to determine the data type from the Datastream interface.

```
data = fetch(Connect, 'ICI') and data = fetch(Connect,  
'ICI', 'P') both return the trailing one-year price time series for the  
given instrument. ('P' is the default value for the 'Field' argument.)
```

```
data = fetch(Connect, 'ICI', {'P', 'PO'}, '08/01/2005')  
returns the closing and opening prices for the given instrument on the  
given date.
```

```
data = fetch(Connect, {'ICI', 'IBM'}, {'P', 'PO'},  
'8/01/2003', '8/01/2005', 'M') returns the monthly closing and  
opening prices for the indicated securities during the specified date  
range.
```

See Also

close, datastream, get, isconnection (Thomson Datastream functions)

get

Purpose Thomson Datastream connection properties

Syntax
value = get(Connect, 'PropertyName')
value = get(Connect)

Arguments

Connect	Thomson Datastream connection object created with the datastream function.
<i>PropertyName</i>	(Optional) A MATLAB string or cell array of strings containing property names. Property names are user datasource endpoint wsdl sources systeminfo version

Description value = get(Connect, 'PropertyName') returns the value of the specified properties for the Thomson Datastream connection object.
value = get(Connect) returns a MATLAB structure where each field name is the name of a property of Connect, and each field contains the value of that property.

See Also close, datastream, fetch, isconnection (Thomson Datastream functions)

Purpose True if valid Thomson Datastream connection

Syntax `x = isconnection(Connect)`

Arguments

Connect	Thomson Datastream connection object created with the datastream function.
---------	--

Description `x = isconnection(Connect)` returns `x = 1` if the connection is a valid Thomson Datastream connection, and `x = 0` if it is not.

Examples

The function

```
c = datastream
```

establishes a connection to the Thomson Datastream API. Then

```
x = isconnection(c)
x = 1
```

indicates that `c` is a valid Thomson Datastream connection.

See Also

`close`, `datastream`, `fetch`, `get` (Thomson Datastream functions)

FactSet

close	Close FactSet connection
close	Close FactSet connection
factset	Connect to FactSet
factset	Connect to FactSet
fetch	Request data from FactSet
fetch	Request data from FactSet
get	FactSet connection properties
get	FactSet connection properties
isconnection	True if valid FactSet connection
isconnection	True if valid FactSet connection

Purpose Close FactSet connection

Syntax `close(Connect)`

Arguments

Connect	FactSet connection object created with the factset function.
---------	--

Description `close(Connect)` closes the connection to the FactSet data server.

See Also `factset`

factset

Purpose Connect to FactSet

Syntax `Connect = factset('UserName', 'SerialNumber', 'Password', 'ID')`

Arguments

UserName	User login name.
SerialNumber	User serial number.
Password	User password.
ID	FactSet customer identification number.

FactSet assigns the values for all of the above input arguments.

Description `Connect = factset('UserName', 'SerialNumber', 'Password', 'ID')` connects to the FactSet FAST interface.

Examples

```
Connect = factset('username', '1234', 'password', 'fsid')
Connect =

    user: 'username'
    serial: '1234'
    password: 'password'
    cid: 'fsid'
```

See Also `close`, `fetch`, `get`, `isconnection` (FactSet functions)

Purpose

Request data from FactSet

Syntax

```
data = fetch(Connect)
data = fetch(Connect, 'Library')
data = fetch(Connect, 'Security', 'Fields')
data = fetch(Connect, 'Security', 'Fields', 'Date')
data = fetch(Connect, 'Security', 'Fields', 'FromDate',
            'ToDate')
data = fetch(Connect, 'Security', 'FromDate', 'ToDate',
            'Period')
```

Arguments

Connect	FactSet connection object created with the factset function.
Library	FactSet formula library.
Security	A MATLAB string or cell array of strings containing the names of securities in a format recognizable by the FactSet server.
<i>Fields</i>	A MATLAB string or cell array of strings indicating the data fields for which data is to be retrieved.
Date	Date string or serial date number indicating date for the requested data. If today's date is entered, yesterday's data is returned.
FromDate	Beginning date for date range.

Note Dates can be specified in any of the formats supported by `datestr` and `datenum` that show a year, month and day.

fetch

<code>ToDate</code>	End date for date range.
<code>Period</code>	Period within date range. <i>Period</i> values are 'd': daily values 'b': business day daily values 'm': monthly values 'mb': beginning monthly values 'me': ending monthly values 'q': quarterly values 'qb': beginning quarterly values 'qe': ending quarterly values 'y': annual values 'yb': beginning annual values 'ye': ending annual values

Description

`data = fetch(Connect)` returns the names of all available formula libraries.

`data = fetch(Connect, 'Library')` returns the valid field names for a given formula library.

`data = fetch(Connect, 'Security', 'Fields')` returns data for the specified security and fields.

`data = fetch(Connect, 'Security', 'Fields', 'Date')` returns security data for the specified fields on the requested date.

`data = fetch(Connect, 'Security', 'Fields', 'FromDate', 'ToDate')` returns security data for the specified fields for the date range `FromDate` to `ToDate`.

`data = fetch(Connect, 'Security', 'FromDate', 'ToDate', 'Period')` returns security data for the date range `FromDate` to `ToDate` with the indicated period.

Examples

Example 1: Obtain the names of the available formula libraries.

```
D = fetch(Connect)
```

Example 2: Obtain the valid field names for the FactSetSecurityCalcs library.

```
D = fetch(Connect, 'fs')
```

Example 3: Obtain closing price of a given security.

```
D = fetch(Connect, 'ABC', 'price')
```

Example 4: Obtain the closing price for the given dates for a given security using the default period of the data.

```
D = fetch(C, 'ABC', 'price', '8/01/99', '8/10/99')
```

Example 5: Obtain the monthly closing price for the given dates for a given security.

```
D = fetch(C, 'ABC', 'price', '8/01/99', '8/10/99', 'm')
```

See Also

close, factset, get, isconnection (FactSet functions)

get

Purpose FactSet connection properties

Syntax
`value = get(Connect, 'PropertyName')`
`value = get(Connect)`

Arguments

<code>Connect</code>	FactSet connection object created with the factset function.
<code>PropertyName</code>	(Optional) A MATLAB string or cell array of strings containing property names. Property names are user serial password cid

Description

`value = get(Connect, 'PropertyName')` returns the value of the specified properties for the FactSet connection object.

`value = get(Connect)` returns a MATLAB structure where each field name is the name of a property of Connect, and each field contains the value of that property.

Examples Use the factset function to establish a connection to FactSet.

```
Connect = factset('Fast_User', '1234', 'Fast_Pass', 'userid')
```


Now use the get function to retrieve the connection property value.

```
h = get(Connect)

h=
    user: 'Fast_User'
  serial: '1234'
password: 'Fast_Pass'
   cid: 'userid'

get(Connect, 'user')

ans =

Fast_User
```

See Also

close, fetch, factset, isconnection (FactSet functions)

isconnection

Purpose True if valid FactSet connection

Syntax `x = isconnection(Connect)`

Arguments

Connect	FactSet connection object created with the factset function.
---------	--

Description `x = isconnection(Connect)` returns `x = 1` if the connection is a valid FactSet connection, and `x = 0` if it is not.

Examples

The function

```
c = factset
```

establishes a FactSet connection.

Then

```
x = isconnection(c);
```

```
x = 1
```

indicates that `c` is a valid FactSet connection.

See Also

`close`, `fetch`, `factset`, `get` (FactSet functions)

Hyperfeed

close	Close Hyperfeed connection
close	Close Hyperfeed connection
fetch	Request data from Hyperfeed
fetch	Request data from Hyperfeed
get	Hyperfeed connection properties
get	Hyperfeed connection properties
hyperfeed	Connect to Hyperfeed
hyperfeed	Connect to Hyperfeed
isconnection	True if valid Hyperfeed connection
isconnection	True if valid Hyperfeed connection

close

Purpose	Close Hyperfeed connection		
Syntax	<code>close(Connect)</code>		
Arguments	<table><tr><td>Connect</td><td>Hyperfeed connection object created with the hyperfeed function.</td></tr></table>	Connect	Hyperfeed connection object created with the hyperfeed function.
Connect	Hyperfeed connection object created with the hyperfeed function.		
Description	<code>close(Connect)</code> closes the connection to the Hyperfeed data server.		
See Also	hyperfeed		

Purpose Request data from Hyperfeed

Syntax

```

data = fetch(Connect, 'Security')
data = fetch(Connect, 'Security', 'Fields')
data = fetch(Connect, 'Security', 'Date')
data = fetch(Connect, 'Security', 'Fields', 'Date')
data = fetch(Connect, 'Security', 'FromDate', 'ToDate')
data = fetch(Connect, 'Security', 'Fields', 'FromDate',
    'ToDate')
data = fetch(Connect, 'Security', 'FromDate', 'ToDate',
    'Period')

```

Arguments

Connect	Hyperfeed connection object created with the hyperfeed function.
Security	A MATLAB string or cell array of strings containing the names of a securities in a format recognizable by the Hyperfeed server.
<i>Fields</i>	A MATLAB string or cell array of strings indicating the data fields for which data is to be retrieved. Some possible values are Symbol Last Date Time Change Open High Low Volume
Date	Date string or serial date number indicating date for the requested data. If today's date is entered, yesterday's data is returned.

`FromDate` Beginning date for historical data.

Note Dates can be specified in any of the formats supported by `datestr` and `datetime` that show a year, month and day.

`ToDate` End date for historical data.

`Period` Period within date range. *Period* values are

'd': daily
'w': weekly
'm': monthly
'v': dividends

Description

`data = fetch(Connect, 'Security')` returns data for all fields from Hyperfeed's Web site for the indicated securities.

`data = fetch(Connect, 'Security', 'Fields')` returns data for the specified fields.

`data = fetch(Connect, 'Security', 'Date')` returns all security data for the requested date.

`data = fetch(Connect, 'Security', 'Fields', 'Date')` returns security data for the specified fields on the requested date.

`data = fetch(Connect, 'Security', 'FromDate', 'ToDate')` returns security data for the date range `FromDate` to `ToDate`.

`data = fetch(Connect, 'Security', 'Fields', 'FromDate', 'ToDate')` returns security data for the specified fields for the date range `FromDate` to `ToDate`.

`data = fetch(Connect, 'Security', 'FromDate', 'ToDate', 'Period')` returns security data for the date range `FromDate` to `ToDate` with the indicated period.

Examples

Obtain the closing price for Coca Cola on April 6, 2000.

```
c = hyperfeed('History');  
  
ClosePrice = fetch(c, 'ko', 'Close', 'Apr 6 00')  
  
ClosePrice =  
  
          730582.00          45.75
```

See Also

close, get, hyperfeed, isconnection (Hyperfeed functions)

get

Purpose Hyperfeed connection properties

Syntax
`value = get(Connect, 'PropertyName')`
`value = get(Connect)`

Arguments

<code>Connect</code>	Hyperfeed connection object created with the hyperfeed function.
<code>PropertyName</code>	(Optional) A MATLAB string or cell array of strings containing property names. Property names are Connection IPAddress Port Socket Version

Description

`value = get(Connect, 'PropertyName')` returns the value of the specified properties for the Hyperfeed connection object.

`value = get(Connect)` returns a MATLAB structure where each field name is the name of a property of `Connect`, and each field contains the value of that property.

Examples Use the hyperfeed function to establish a connection to Hyperfeed.

```
c = hyperfeed('Price')
```

Now use the `get` function to retrieve the connection property value.

```
h = get(c, Connection)
```

```
h=
```

```
connection: 3  
table: 'Price'
```


See Also close, fetch, hyperfeed, isconnection (Hyperfeed functions)

hyperfeed

Purpose	Connect to Hyperfeed		
Syntax	Connect = hyperfeed(<i>Table</i>)		
Arguments	<table><tr><td><i>Table</i></td><td>A MATLAB string indicating the Hyperfeed table (database) to access. Possible values are Price (Default) Profile History</td></tr></table>	<i>Table</i>	A MATLAB string indicating the Hyperfeed table (database) to access. Possible values are Price (Default) Profile History
<i>Table</i>	A MATLAB string indicating the Hyperfeed table (database) to access. Possible values are Price (Default) Profile History		
Description	Connect = hyperfeed(<i>Table</i>) connects to the indicated Hyperfeed table.		
Examples	<pre>c = hyperfeed('Price')</pre> <p>connects to the Hyperfeed Price table.</p>		
See Also	close, fetch, get, isconnection (Hyperfeed functions)		

Purpose	True if valid Hyperfeed connection		
Syntax	<code>x = isconnection(Connect)</code>		
Arguments	<table><tr><td>Connect</td><td>Hyperfeed connection object created with the hyperfeed function.</td></tr></table>	Connect	Hyperfeed connection object created with the hyperfeed function.
Connect	Hyperfeed connection object created with the hyperfeed function.		
Description	<code>x = isconnection(Connect)</code> returns <code>x = 1</code> if the connection is a valid Hyperfeed connection, and <code>x = 0</code> if it is not.		
Examples	<p>The function</p> <pre>c = hyperfeed</pre> <p>establishes a Hyperfeed connection, <code>c</code>, to the Price table.</p> <p>Then</p> <pre>x = isconnection(c);</pre> <pre>x = 1</pre> <p>indicates that <code>c</code> is a valid Hyperfeed connection.</p>		
See Also	<code>close</code> , <code>fetch</code> , <code>get</code> , <code>hyperfeed</code> (Hyperfeed functions)		

FT Interactive Data

close	Close FT Interactive Data connection
close	Close FT Interactive Data connection
fetch	Request data from FT Interactive Data
fetch	Request data from FT Interactive Data
get	FT Interactive Data connection properties
get	FT Interactive Data connection properties
idc	Connect to FT Interactive Data
idc	Connect to FT Interactive Data
isconnection	True if valid FT Interactive Data connection
isconnection	True if valid FT Interactive Data connection

Purpose	Close FT Interactive Data connection		
Syntax	<code>close(Connect)</code>		
Arguments	<table><tr><td><code>Connect</code></td><td>FT Interactive Data connection object created with the <code>idc</code> function.</td></tr></table>	<code>Connect</code>	FT Interactive Data connection object created with the <code>idc</code> function.
<code>Connect</code>	FT Interactive Data connection object created with the <code>idc</code> function.		
Description	<code>close(Connect)</code> closes the connection to the FT Interactive Data server.		
Examples	<pre>c = idc establishes an FT Interactive Data connection, c. close(c) closes this connection.</pre>		
See Also	<code>idc</code>		

fetch

Purpose Request data from FT Interactive Data

Syntax

```
data = fetch(Connect, 'Security', 'Fields')
data = fetch(Connect, 'Security', 'Fields', 'FromDate',
             'ToDate')
data = fetch(Connect, 'Security', 'Fields', 'FromDate',
             'ToDate', 'Period')
data = fetch(Connect, '', 'GUILookup', 'GUICategory')
```

Arguments

Connect	FT Interactive Data connection object created with the <code>idc</code> function.
Security	A MATLAB string containing the name of a security in a format recognizable by the FT Interactive Data server.
<i>Fields</i>	A MATLAB string or cell array of strings indicating specific fields for which data is to be provided. Valid field names are in the file <code>@idc/idcfields.mat</code> . The variable <code>bbfieldnames</code> contains the list of field names.
FromDate	Beginning date for historical data.
<hr/> Note Dates can be specified in any of the formats supported by <code>datestr</code> and <code>datenum</code> that show a year, month and day. <hr/>	
ToDate	End date for historical data.
<i>Period</i>	Period within date range.
<i>GUICategory</i>	GUI category. Possible values are F (All valid field categories) S (All valid security categories)

Description

`data = fetch(Connect, 'Security', 'Fields')` returns data for the indicated fields of the designated securities. Load the file `idc/idcfields` to see the list of supported fields.

`data = fetch(Connect, 'Security', 'Fields', 'FromDate', 'ToDate')` returns historical data for the indicated fields of the designated securities.

`data = fetch(Connect, 'Security', 'Fields', 'FromDate', 'ToDate', 'Period')` returns historical data for the indicated fields of the designated securities with the designated dates and period. Consult the Remote Plus documentation for a list of valid *Period* values.

`data = fetch(Connect, '', 'GUILookup', 'GUICategory')` opens the FT Interactive Data dialog box for selecting fields or securities.

Examples

```
D = fetch(Connect, '', 'GUILOOKUP', 'S')
```

opens the dialog box for looking up securities.

```
D = fetch(Connect, '', 'GUILOOKUP', 'F')
```

opens the dialog box for selecting fields.

See Also

`close`, `get`, `idc`, `isconnection` (FT Interactive Data functions)

get

Purpose FT Interactive Data connection properties

Syntax `value = get(Connect, 'PropertyName')`
`value = get(Connect)`

Arguments

<code>Connect</code>	FT Interactive Data connection object created with the <code>idc</code> function.
<code>PropertyName</code>	(Optional) A MATLAB string or cell array of strings containing property names. Property names are <code>Connected</code> <code>Connection</code> <code>Queued</code>

Description `value = get(Connect, 'PropertyName')` returns the value of the specified properties for the FT Interactive Data connection object. `'PropertyName'` is a string or cell array of strings containing property names.

`value = get(Connect)` returns a MATLAB structure. Each field name is the name of a property of `Connect`, and each field contains the value of that property.

See Also `close`, `fetch`, `idc`, `isconnection` (FT Interactive Data functions)

Purpose	Connect to FT Interactive Data
Syntax	<code>Connect = idc</code>
Description	<code>Connect = idc</code> connects to the FT Interactive Data server. <code>Connect</code> is a connection handle used by other functions to obtain data.
Examples	<pre>c = idc</pre> <p>makes a connection to the FT Interactive Data server.</p>
See Also	<code>close</code> , <code>fetch</code> , <code>get</code> , <code>isconnection</code> (FT Interactive Data functions)

isconnection

Purpose True if valid FT Interactive Data connection

Syntax `x = isconnection(Connect)`

Arguments Connect FT Interactive Data connection object created with the `idc` function.

Description `x = isconnection(Connect)` returns `x = 1` if the connection is a valid FT Interactive Data connection, and `x = 0` if it is not.

Examples The function

```
c = idc
```

establishes an FT Interactive Data connection, `c`.

Then

```
x = isconnection(c)
x = 1
```

indicates that `c` is a valid FT Interactive Data connection.

See Also `close`, `fetch`, `get`, `idc` (FT Interactive Data functions)

Federal Reserve Economic Data

close	Close FRED connection
close	Close FRED connection
fetch	Request data from FRED
fetch	Request data from FRED
fred	Connect to FRED
fred	Connect to FRED
get	FRED connection properties
get	FRED connection properties
isconnection	True if valid FRED connection
isconnection	True if valid FRED connection

fred

Purpose Connect to FRED

Syntax
Connect = fred(URL)
Connect = fred

Arguments URL Create a connection using a specified URL.

Description Connect = fred(URL) establishes a connection to a FRED data server.
Connect = fred verifies that the URL
`http://research.stlouisfed.org/fred2/series` is accessible and
creates a connection.

Examples
`c = fred(http://research.stlouisfed.org/fred2/series)`
makes a connection to the FRED server at the specified URL.

See Also close, fetch, get, isconnection

Purpose Close FRED connection

Syntax `close(Connect)`

Arguments

<code>Connect</code>	FRED connection object created with the fred function.
----------------------	--

Description `close(Connect)` closes the connection to the FRED data server.

Examples

```
c = fred(http://research.stlouisfed.org/fred2/series)
```

establishes a FRED connection, c.

```
close(c)
```

closes this connection.

See Also fred

fetch

Purpose Request data from FRED

Syntax

```
data = fetch(Connect, 'Security')
data = fetch(Connect, 'Security', 'D1')
data = fetch(Connect, 'Security', 'D1', 'D2')
```

Arguments

Connect	FRED connection object created with the fred function.
Security	MATLAB string containing the name of a security in a format recognizable by the FRED server.
'D1'	MATLAB string or date number indicating the date from which data is to be retrieved. Returns all data for the date 'D1'; if 'D1' is today's date, the data from yesterday will be returned.
'D2'	MATLAB string or date number indicating the date range from which data is to be retrieved. Returns all data for the given security for the date range 'D1' to 'D2'.

Note You can specify dates in any of the formats supported by datestr and datenum that show a year, month, and day.

Description For a given security, fetch returns historical data using the FRED connection.

Examples Fetch all available daily U.S. to Euro foreign exchange rates:

```
d = fetch(f, 'DEXUSEU')
```

```
d =
    Title: ' U.S. / Euro Foreign Exchange Rate'
    SeriesID: ' DEXUSEU'
    Source: ' Board of Governors of the Federal Reserve System'
    Release: ' H.10 Foreign Exchange Rates'
    SeasonalAdjustment: ' Not Applicable'
    Frequency: ' Daily'
    Units: ' U.S. Dollars to One Euro'
    DateRange: ' 1999-01-04 to 2006-06-19'
    LastUpdated: ' 2006-06-20 9:39 AM CT'
    Notes: ' Noon buying rates in New York City for cable transfers payable'
    Data: [1877x2 double]
```

Fetch data for a given date range:

```
d = fetch(f, 'DEXUSEU', '01/01/2006', '06/01/2006')
d =
```

```
    Title: ' U.S. / Euro Foreign Exchange Rate'
    SeriesID: ' DEXUSEU'
    Source: ' Board of Governors of the Federal Reserve System'
    Release: ' H.10 Foreign Exchange Rates'
    SeasonalAdjustment: ' Not Applicable'
    Frequency: ' Daily'
    Units: ' U.S. Dollars to One Euro'
    DateRange: ' 1999-01-04 to 2006-06-19'
    LastUpdated: ' 2006-06-20 9:39 AM CT'
    Notes: ' Noon buying rates in New York City for cable transfers payable'
    Data: [105x2 double]
```

See Also

close, get, isconnection

get

Purpose FRED connection properties

Syntax

```
value = get(Connect, 'PropertyName')  
value = get(Connect)
```

Arguments

Connect	FRED connection object created with the fred function.
'PropertyName'	A MATLAB string or cell array of strings containing property names. Property names are: <ul style="list-style-type: none">• url• ip• port

Description

`value = get(Connect, 'PropertyName')` returns a MATLAB structure containing the value of the specified properties for the FRED connection object.

`value = get(Connect)` returns the value for all properties.

Examples

```
c = FRED(http://research.stlouisfed.org/fred2/series)
```

establishes a FRED connection, c.

The syntax

```
p = get(c, {'port', 'ip'})
```


returns

```
p =  
  port: 8194  
  ip: 111.222.33.444
```

See Also

close, fetch, isconnection

isconnection

Purpose True if valid FRED connection

Syntax `x = isconnection(Connect)`

Arguments

<code>Connect</code>	FRED connection object created with the <code>fred</code> function.
----------------------	---

Description `x = isconnection(Connect)` returns `x = 1` if the connection is a valid FRED connection, and `x = 0` if it is not.

Examples The function

```
c = fred(http://research.stlouisfed.org/fred2/series)
```

establishes a FRED connection, `c`.

Then

```
x = isconnection(c)
```

```
x = 1
```

indicates that `c` is a valid FRED connection.

See Also `close`, `fetch`, `get`

Yahoo!

close	Close Yahoo! connection
close	Close Yahoo! connection
fetch	Request data from Yahoo!
fetch	Request data from Yahoo!
get	Yahoo! connection properties
get	Yahoo! connection properties
isconnection	True if valid Yahoo! connection
isconnection	True if valid Yahoo! connection
yahoo	Connect to Yahoo!
yahoo	Connect to Yahoo!

close

Purpose	Close Yahoo! connection		
Syntax	<code>close(Connect)</code>		
Arguments	<table><tr><td><code>Connect</code></td><td>Yahoo! connection object created with the yahoo function.</td></tr></table>	<code>Connect</code>	Yahoo! connection object created with the yahoo function.
<code>Connect</code>	Yahoo! connection object created with the yahoo function.		
Description	<code>close(Connect)</code> closes the connection to the Yahoo! data server.		
See Also	yahoo		

Purpose

Request data from Yahoo!

Syntax

```
data = fetch(Connect, 'Security')
data = fetch(Connect, 'Security', 'Fields')
data = fetch(Connect, 'Security', 'Date')
data = fetch(Connect, 'Security', 'Fields', 'Date')
data = fetch(Connect, 'Security', 'FromDate', 'ToDate')
data = fetch(Connect, 'Security', 'Fields', 'FromDate',
    'ToDate')
data = fetch(Connect, 'Security', 'FromDate', 'ToDate',
    'Period')
```

Arguments

Connect	Yahoo! connection object created with the yahoo function.
Security	A MATLAB string or cell array of strings containing the name of a security in a format recognizable by the Yahoo! server.

Note Retrieving multiple securities at one time is not supported for Yahoo. You must fetch a single security at a time.

<i>Fields</i>	<p>A MATLAB string or cell array of strings indicating the data fields for which data is to be retrieved. A partial list of supported values for current market data are</p> <ul style="list-style-type: none"> Symbol Last Date Time Change Open High
---------------	---

fetch

	Low
	Volume
	A partial list of supported values for historical data are
	Close
	Date
	High
	Low
	Open
	Volume
	Adj. Close*
	For a complete list of supported values for market and historical data, see <code>yhfields.mat</code> .
Date	Date string or serial date number indicating date for the requested data. If today's date is entered, yesterday's data is returned.
FromDate	Beginning date for historical data.
<hr/> Note Dates can be specified in any of the formats supported by <code>datestr</code> and <code>datenum</code> that show a year, month and day. <hr/>	
ToDate	End date for historical data.
Period	Period within date range. <i>Period</i> values are 'd': daily 'w': weekly 'm': monthly 'v': dividends

Description

`data = fetch(Connect, 'Security')` returns data for all fields from Yahoo!'s Web site for the indicated security. Retrieving multiple

securities at one time is not supported for Yahoo. You must fetch a single security at a time

```
data = fetch(Connect, 'Security', 'Fields') returns data for
the specified fields.
```

```
data = fetch(Connect, 'Security', 'Date') returns all security
data for the requested date.
```

```
data = fetch(Connect, 'Security', 'Fields', 'Date') returns
security data for the specified fields on the requested date.
```

```
data = fetch(Connect, 'Security', 'FromDate', 'ToDate')
returns security data for the date range FromDate to ToDate.
```

```
data = fetch(Connect, 'Security', 'Fields', 'FromDate',
'ToDate') returns security data for the specified fields for the date
range FromDate to ToDate.
```

```
data = fetch(Connect, 'Security', 'FromDate', 'ToDate',
'Period') returns security data for the date range FromDate to ToDate
with the indicated period.
```

Examples

Example 1: Obtain the closing price for Coca Cola on April 6, 2000.

```
c = yahoo;

ClosePrice = fetch(c, 'ko', 'Close', 'Apr 6 00')

ClosePrice =

          730582.00          45.75
```

Example 2: Use the Yahoo! data server to obtain the last prices for a set of equities.

```
y = yahoo;

FastFood = fetch(y, {'ko', 'pep', 'mcd'}, 'Last')
```

fetch

```
FastFood =  
    Last: [3x1 double]
```

```
FastFood.Last
```

```
ans =
```

```
    42.96  
    45.71  
    23.70
```

See Also

close, get, isconnection , yahoo (Yahoo! functions)

Purpose	Yahoo! connection properties	
Syntax	<pre>value = get(Connect, 'PropertyName') value = get(Connect)</pre>	
Arguments	Connect	Yahoo! connection object created with the yahoo function.
	<i>PropertyName</i>	(Optional) A MATLAB string or cell array of strings containing property names. Currently the only property name recognized is url.
Description	<p><code>value = get(Connect, 'PropertyName')</code> returns the value of the specified properties for the Yahoo! connection object.</p> <p><code>value = get(Connect)</code> returns a MATLAB structure where each field name is the name of a property of Connect, and each field contains the value of that property.</p>	
Examples	<p>Use the yahoo function to establish a connection to Yahoo!.</p> <pre>c = yahoo c = url: 'http://quote.yahoo.com'</pre> <p>Now use the get function to retrieve the connection property value.</p> <pre>get(c, 'url') ans = url: 'http://quote.yahoo.com'</pre>	
See Also	close, fetch, isconnection, yahoo (Yahoo! functions)	

isconnection

Purpose True if valid Yahoo! connection

Syntax `x = isconnection(Connect)`

Arguments Connect Yahoo! connection object created with the yahoo function.

Description `x = isconnection(Connect)` returns `x = 1` if the connection is a valid Yahoo! connection, and `x = 0` if it is not.

Examples The function

```
c = yahoo
```

establishes a Yahoo! connection, c.

Then

```
x = isconnection(c)
x = 1
```

indicates that c is a valid Yahoo! connection.

See Also `close`, `fetch`, `get`, `yahoo` (Yahoo! functions)

Purpose

Connect to Yahoo!

Syntax

```
Connect = yahoo
Connect = yahoo('URL', 'IPAddress', PortNumber)
```

Arguments

URL	Must be 'http://quote.yahoo.com'.
IPAddress	A MATLAB string containing the Internet address of proxy server machine.
PortNumber	Port number on proxy server.

Description

Connect = yahoo verifies that the URL http://quote.yahoo.com is accessible and creates a connection handle.

Connect = yahoo('URL', 'IPAddress', PortNumber) connects to Yahoo! through a proxy server using the IP address and port number provided. This form of the yahoo function may be required when connecting to Yahoo! from behind an internal firewall.

Examples

Use the yahoo function to establish a connection to the Yahoo! data server.

```
Connect = yahoo

Connect =

    url: 'http://quote.yahoo.com'
```

Use the yahoo function to establish a connection to the Yahoo! data server, providing an IP address and port number on a proxy server.

```
Connect = yahoo('http://quote.yahoo.com', '111.222.33.444', 5678)

Connect =

    url: 'http://quote.yahoo.com'
```

```
ip: '111.222.33.444'  
port: 5678
```

See Also

close, fetch, get, isconnection (Yahoo! functions)

Functions — By Category

Bloomberg (p. 3-2)

Get Bloomberg financial data

Thomson Datastream (p. 3-22)

Get Thomson Datastream financial data

FactSet (p. 3-30)

Get FactSet financial data

Hyperfeed (p. 3-39)

Get Hyperfeed financial data

FT Interactive Data (p. 3-48)

Get FT Interactive Data financial data

Federal Reserve Economic Data
(p. 3-55)

Get Federal Reserve Economic Data
financial data

Yahoo! (p. 3-63)

Get Yahoo! financial data

Bloomberg

bloomberg	Connect to Bloomberg
bloomberg	Connect to Bloomberg
close	Close Bloomberg connection
close	Close Bloomberg connection
fetch	Request data from Bloomberg
fetch	Request data from Bloomberg
get	Bloomberg connection properties
get	Bloomberg connection properties
isconnection	True if valid Bloomberg connection
isconnection	True if valid Bloomberg connection
pricevol	Price and volume (demonstration)
pricevol	Price and volume (demonstration)
showtrades	Recent trade data (demonstration)
showtrades	Recent trade data (demonstration)
stockticker	Trades with volumes (demonstration)
stockticker	Trades with volumes (demonstration)

Purpose	Connect to Bloomberg	
Syntax	<code>Connect = bloomberg(PortNumber, 'IPAddress')</code> <code>Connect = bloomberg</code>	
Arguments	PortNumber	Port on machine where connection is being made.
	IPAddress	A MATLAB string containing the Internet address of machine where connection is being made.
Description	<code>Connect = bloomberg(PortNumber, 'IPAddress')</code> establishes a connection to a Bloomberg data server using the port number, PortNumber, and the Internet address, IPAddress. <code>Connect = bloomberg</code> establishes a connection to a Bloomberg data server using port number 8194 and the default Internet address provided when the Bloomberg software was installed on your machine.	
Examples	<code>c = bloomberg(8194, '111.222.33.444')</code> makes a connection to the Bloomberg server on port 8194 of the machine with Internet address 111.222.33.444.	
See Also	<code>close</code> , <code>fetch</code> , <code>get</code> , <code>isconnection</code> (Bloomberg functions)	

close

Purpose	Close Bloomberg connection		
Syntax	<code>close(Connect)</code>		
Arguments	<table><tr><td><code>Connect</code></td><td>Bloomberg connection object created with the bloomberg function.</td></tr></table>	<code>Connect</code>	Bloomberg connection object created with the bloomberg function.
<code>Connect</code>	Bloomberg connection object created with the bloomberg function.		
Description	<code>close(Connect)</code> closes the connection to the Bloomberg data server.		
Examples	<pre>c = bloomberg(8194, '111.222.33.444')</pre> <p>establishes a Bloomberg connection, <code>c</code>.</p> <pre>close(c)</pre> <p>closes this connection.</p>		
See Also	<code>bloomberg</code>		

Purpose

Request data from Bloomberg

Syntax

```
data = fetch(Connect, 'Security')
data = fetch(Connect, 'Security', 'HEADER', 'Flag', 'Ident')
data = fetch(Connect, 'Security', 'GETDATA', 'Fields',
'Override', 'Ident', 'Values')
data = fetch(Connect, 'Security', 'TIMESERIES', 'Date',
'Minutes', 'TickField')
data = fetch(Connect, 'Security', 'HISTORY', 'Fields',
'FromDate', 'ToDate', 'Period', 'Currency', 'Ident')
ticker = fetch(Connect, 'SearchString', 'LOOKUP', 'Market')
data = fetch(Connect, 'Security', 'REALTIME', 'Fields',
MATLABProg)
data = fetch(Connect, Security, 'STOP')
```

Arguments

Connect	Bloomberg connection object created with the bloomberg function.
Security	A MATLAB string containing the name of a security in a format recognizable by the Bloomberg server. You can substitute a CUSIP number for a security name if you want.

Note The Security argument may be a cell array of strings containing a list of securities.

<i>Flag</i>	A MATLAB string indicating the dates from which data is to be retrieved. Possible values are DEFAULT: Data from most recent bid, ask, or trade. If a <i>Flag</i> value is not specified, 'DEFAULT' is assumed. TODAY: Today's data only. ENHANCED: Data from most recent date of each individual field.
<i>Currency</i>	(Optional) Currency in which historical returns are provided. Valid currencies are listed in the file @bloomberg/bbfields.mat. Default = [].
<i>Ident</i>	(Optional) Security type identifier. Valid security type identifiers are listed in the file @bloomberg/bbfields.mat. Default = [].
<i>Fields</i>	A MATLAB string or cell array of strings indicating specific fields for which data is requested. Valid field names are listed in the file @bloomberg/bbfields.mat. The variable bbfieldnames contains the list of field names. Default = [].
<i>Override</i>	(Optional) String or cell array of strings containing override field list. Default = [].
<i>Values</i>	(Optional) String or cell array of strings containing override field values.
<i>Date</i>	Date string or serial date number indicating date for the time series. Specify now for today's time series data.
<i>Minutes</i>	(Optional) Tick interval in minutes.
<i>TickField</i>	(Optional) The field can be specified as a string or numeric value (e.g., TickField = 'Trade' or TickField = 1 return data for ticks of type Trade. Use the function dftool('ticktypes') to return the list of intraday tick fields.

`FromDate` Beginning date for historical data.

Note Dates can be specified in any of the formats supported by `datestr` and `datenum` that show a year, month and day.

`ToDate` End date for historical data.

`Period` (Optional) Period of the data. A MATLAB three-part string with the format:

'Frequency Days Data'

Frequency Values

d: daily (default)

w: weekly

m: monthly

q: quarterly

y: yearly

Days Values

o: omit all days for which there is no data (default)

i: include all trading days

a: include all calendar days

Data Values

b: report missing data using Bloomberg (default)

s: show missing data as last found value

n: report missing data as NaN

For example, 'dan' returns daily data for all calendar days, reporting missing values as NaN. If a value is unspecified (e.g., ' n'), the unspecified values are replaced by defaults.

Note If *Period* is not specified, default values are used.

<i>Currency</i>	(Optional) Currency type. The file @bloomberg/bbfields.mat lists the supported currencies.
<i>Market</i>	A MATLAB string indicating the market in which a particular security trades. <i>Market</i> values are Comdty: (Commodities) Corp: (Corporate bonds) Equity: (Equities) Govt: (Government bonds) Index: (Indexes) M-Mkt: (Money Market securities) Mtge: Mortgage-backed securities) Muni: (Municipal bonds) Pfd: (Preferred stocks)
MATLABProg	A string that is the name of any valid MATLAB program.

Description

For a given security, `fetch` returns header (default), current, time series, real time, and historical data via the Bloomberg connection.

`data = fetch(Connect, 'Security')` fills the header fields with data from the most recent date with a bid, ask, or trade.

`data = fetch(Connect, 'Security', 'HEADER', 'Flag', 'Ident')` returns data for the most recent date of each individual field for the specified security type identifiers, based upon the value of *Flag*.

- If *Flag* is DEFAULT, `fetch` fills the header fields with data from the most recent date with a bid, ask, or trade. This is the equivalent of `data = fetch(Connect, 'Security')`.
- If *Flag* is TODAY, `fetch` returns the header field data with data from today only.
- If *Flag* is ENHANCED, `fetch` returns the header field data for the most recent date of each individual field. In this case, for example, the bid and ask group fields could come from different dates.

`data = fetch(Connect, 'Security', 'GETDATA', 'Fields', 'Override', 'Ident', 'Values')` returns the current market data for the specified fields of the indicated security. You can further specify the data with the optional *Override*, *Values* and *Ident* arguments.

`data = fetch(Connect, 'Security', 'TIMESERIES', 'Date', 'Minutes', 'TickField')` returns the tick data for a security for the specified date. You can further specify the data with the optional *Minutes* and *TickField* arguments.

You can specify *TickField* as a string or numeric value, e.g., `TickField = 'Trade'` or `TickField = 1` returns data for ticks of type *Trade*. The function `dftool('ticktypes')` returns the list of intraday tick fields. Intraday tick data requested with an interval is returned with the columns representing

- Time
- Open
- High
- Low
- Value of last tick
- Volume total value of ticks
- Total value of ticks for the time range
- Number of ticks

Columns 7 and 8 are provided only if they make sense for the requested field.

For today's tick data, specify

```
data = fetch(Connect, 'Security', 'TIMESERIES', now)
```

For today's trade time series aggregated into five-minute intervals, enter

```
data = fetch(Connect, 'Security', 'TIMESERIES', ...  
now, 5, 'Trade')
```

`data = fetch(Connect, 'Security', 'HISTORY', 'Fields', 'FromDate', 'ToDate', 'Period', 'Currency', 'Ident')` returns historical data for the specified field for the date range `FromDate` to `ToDate`. You can further specify the date range by setting the time period with the optional *Period* argument. You can further specify the data to be returned by appending the *Currency* or *Ident* argument.

`ticker = fetch(Connect, 'SearchString', 'LOOKUP', 'Market')` uses `SearchString` to find the ticker symbol for a security trading in a designated market. The output `ticker` is a column vector of possible ticker values.

Note If you supply *Ident* without a period or currency, enter `[]` for the missing values.

`data = fetch(Connect, 'Security', 'REALTIME', 'Fields', 'MATLABProg)` subscribes to a given security or list of securities, requesting the indicated fields, and runs any specified MATLAB function. See `pricevol`, `showtrades`, or `stockticker` for information on the data returned by asynchronous Bloomberg events.

`data = fetch(Connect, Security, 'STOP')` unsubscribes the list of securities from processing Bloomberg real-time events.

Examples

Returning Header Data

```
D = fetch(C, 'ABC US Equity')
```

returns the header data for a United States equity with ticker ABC.

Opening and Closing Prices

```
D = fetch(C, 'ABC US Equity', 'GETDATA', ...  
        {'Last_Price'; 'Open'})
```

returns the opening and closing prices.

Override Fields

```
D = fetch(C, '3358ABCD4 Corp', 'GETDATA', ...  
        {'YLD_YTM_ASK', 'ASK', 'OAS_SPREAD_ASK', 'OAS_VOL_ASK'}, ...  
        {'PX_ASK', 'OAS_VOL_ASK'}, {'99.125000', '14.000000'})
```

returns the requested fields given override fields and values.

Time Series

```
D = fetch(C, 'ABC US Equity', 'TIMESERIES', now)
```

return today's time series.

Time Intervals

```
D = fetch(C, 'ABC US Equity', 'TIMESERIES', now, 5, 'Trade')
```

returns today's trade time series for the given security aggregated into five-minute intervals.

Default Closing Price

```
D = fetch(C, 'ABC US Equity', 'HISTORY', 'Last_Price', ...  
        '8/01/99', '8/10/99')
```

returns the closing price for the given dates using the default period of the data.

fetch

Monthly Closing Price

```
D = fetch(C, 'ABC US Equity', 'HISTORY', 'Last_Price', ...  
        '8/01/99', '9/30/00', 'm')
```

returns the monthly closing price for the given dates for the given security.

See Also

bloomberg, close, get, isconnection (Bloomberg functions)

Purpose Bloomberg connection properties

Syntax
`value = get(Connect, 'PropertyName')`
`value = get(Connect)`

Arguments

<code>Connect</code>	Bloomberg connection object created with the bloomberg function.
<code>PropertyName</code>	(Optional) A MATLAB string or cell array of strings containing property names. Property names are Connection IPAddress Port Socket Version

Description `value = get(Connect, 'PropertyName')` returns a MATLAB structure containing the value of the specified properties for the Bloomberg connection object.

`value = get(Connect)` returns the value for all properties.

Examples

```
c = bloomberg(8194, '111.222.33.444')
```

establishes a Bloomberg connection, `c`.

The syntax

```
p = get(c, {'Port', 'IPAddress'})
```

get

returns

```
p =  
  port: 8194  
  ipaddress: 111.222.33.444
```

See Also

bloomberg, close, fetch, isconnection (Bloomberg functions)

Purpose	True if valid Bloomberg connection		
Syntax	<code>x = isconnection(Connect)</code>		
Arguments	<table><tr><td>Connect</td><td>Bloomberg connection object created with the bloomberg function.</td></tr></table>	Connect	Bloomberg connection object created with the bloomberg function.
Connect	Bloomberg connection object created with the bloomberg function.		
Description	<code>x = isconnection(Connect)</code> returns <code>x = 1</code> if the connection is a valid Bloomberg connection, and <code>x = 0</code> if it is not.		
Examples	<p>The function</p> <pre>c = bloomberg(8194, '111.222.33.444')</pre> <p>establishes a Bloomberg connection, <code>c</code>.</p> <p>Then</p> <pre>x = isconnection(c) x = 1</pre> <p>indicates that <code>c</code> is a valid Bloomberg connection.</p>		
See Also	<code>bloomberg</code> , <code>close</code> , <code>fetch</code> , <code>get</code> (Bloomberg functions)		

pricevol

Purpose	Price and volume (demonstration)	
Syntax	pricevol(InputList)	
Arguments	InputList	Fields for which real-time data is sought.

Description pricevol(InputList) demonstrates the Bloomberg real-time data import functionality. InputList is an input list of the elements:

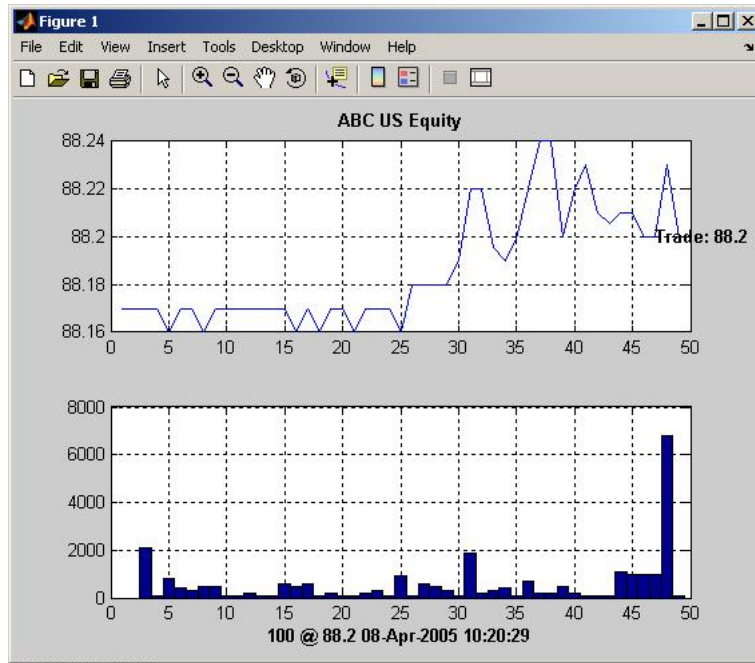
InputList(1) = COM.Bloomberg.Data.1	Bloomberg handle
InputList(2) = 1	Event ID
InputList(3) = ('Security')	Security string
InputList(4) = 1	Cookie
InputList(5) = 2	Field number ID
InputList(6) = {[43.58]}	Return data for the given tick
InputList(7) = 0	Status
InputList(8)	Structure containing the above fields
InputList(9) = 'Data'	Event type

The input argument InputList(8) contains the necessary information to process real-time events.

Examples The following example shows the use of this function.

```
b = bloomberg;  
d = fetch(b, 'ABC US Equity', 'REALTIME', ...  
{ 'Last_Trade', 'Volume' }, 'pricevol');
```

The output displays the most recent Trade and Volume in the figure and shows the most recent trade with volumes.



See Also `showtrades`, `stockticker`

showtrades

Purpose Recent trade data (demonstration)

Syntax showtrades(InputList)

Arguments

InputList	Fields for which real-time data is sought.
-----------	--

Description showtrades(InputList) demonstrates the Bloomberg real-time data import functionality. InputList is an input list of the elements:

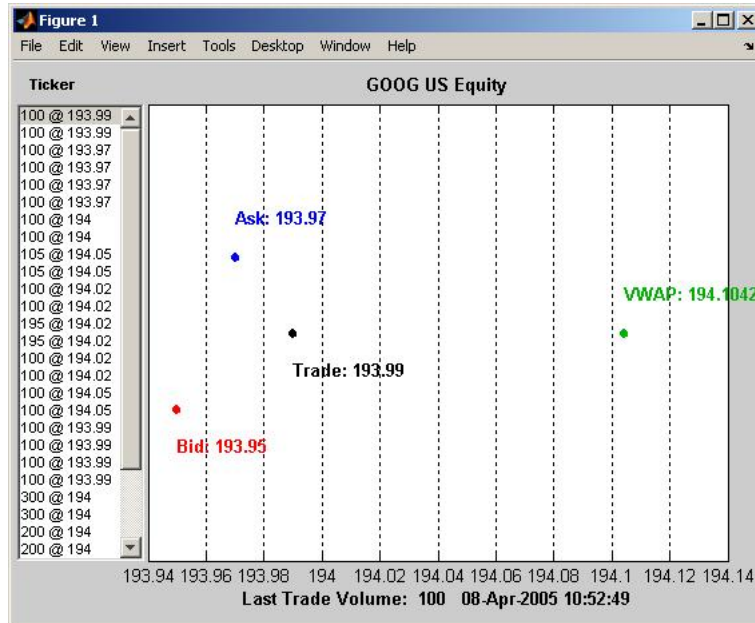
InputList(1) = COM.Bloomberg.Data.1	Bloomberg handle
InputList(2) = 1	Event ID
InputList(3) = ('Security')	Security string
InputList(4) = 1	Cookie
InputList(5) = 2	Field number ID
InputList(6) = {[43.58]}	Return data for the given tick
InputList(7) = 0	Status
InputList(8)	Structure containing the above fields
InputList(9) = 'Data'	Event type

The input argument InputList(8) contains the necessary information to process real-time events.

Examples The following example shows the use of this function.

```
b = bloomberg;  
d = fetch(b, 'GOOG US Equity', 'REALTIME', ...  
{ 'Last_Trade', 'Bid', 'Ask', 'Volume', 'VWAP' }, 'showtrades');
```

The output shows the most recent Trade, Bid, Ask and VWAP (volume-weighted adjusted price) and a list of the most recent trades with volumes.



See Also pricevol, stockticker

stockticker

Purpose Trades with volumes (demonstration)

Syntax stockticker(InputList)

Arguments

InputList Fields for which real-time data is sought.

Description

stockticker(InputList) demonstrates the Bloomberg real-time data import functionality. InputList is an input list of the elements:

InputList(1) = COM.Bloomberg.Data.1	Bloomberg handle
InputList(2) = 1	Event ID
InputList(3) = ('Security')	Security string
InputList(4) = 1	Cookie
InputList(5) = 2	Field number ID
InputList(6) = {[43.58]}	Return data for the given tick
InputList(7) = 0	Status
InputList(8)	Structure containing the above fields
InputList(9) = 'Data'	Event type

The input argument InputList(8) contains the necessary information to process real-time events.

Examples

The following example shows the use of this function. The output provides a list of trades with volumes for each requested security.

```
b = bloomberg;  
d = fetch(b,{'IBM US Equity','EMC US Equity','NTAP US Equity'},...  
'REALTIME',{'Last_Trade','Volume'},'stockticker');  
** EMC US Equity ** 0 @ 12.65 08-Apr-2005 10:24:57
```



```
** IBM US Equity ** 0 @ 88.17 08-Apr-2005 10:24:57
** NTAP US Equity ** 0 @ 29.02 08-Apr-2005 10:24:57
** EMC US Equity ** 200 @ 12.66 08-Apr-2005 10:24:58
** EMC US Equity ** 1400 @ 12.65 08-Apr-2005 10:24:58
** EMC US Equity ** 3100 @ 12.66 08-Apr-2005 10:25:00
** IBM US Equity ** 1300 @ 88.17 08-Apr-2005 10:25:00
.
.
.
```

See Also pricevol, showtrades

Thomson Datastream

close	Close Thomson Datastream connection
close	Close Thomson Datastream connection
datastream	Thomson Datastream API connection
datastream	Thomson Datastream API connection
fetch	Request data from Thomson Datastream
fetch	Request data from Thomson Datastream
get	Thomson Datastream connection properties
get	Thomson Datastream connection properties
isconnection	True if valid Thomson Datastream connection
isconnection	True if valid Thomson Datastream connection

Purpose	Close Thomson Datastream connection		
Syntax	<code>close(Connect)</code>		
Arguments	<table><tr><td>Connect</td><td>Thomson Datastream connection object created with the <code>datastream</code> function.</td></tr></table>	Connect	Thomson Datastream connection object created with the <code>datastream</code> function.
Connect	Thomson Datastream connection object created with the <code>datastream</code> function.		
Description	<code>close(Connect)</code> closes the connection to the Thomson Datastream data server.		
See Also	<code>datastream</code>		

datastream

Purpose	Thomson Datastream API connection	
Syntax	Connect = datastream('UserName', 'Password', 'Source', 'URL')	
Arguments	UserName	User name.
	Password	User password.
	Source	To connect to the Thomson Datastream API, enter 'Datastream' in this field.
	URL	Web URL.

Note Thomson assigns the values you need to enter for each argument. Enter all arguments as MATLAB strings.

Description Connect = datastream('UserName', 'Password', 'Source', 'URL') makes a connection to the Thomson Datastream API, which provides access to the Thomson Datastream content.

Examples Use the datastream function to establish a connection to the Thomson Datastream API, which provides you access to the Thomson Datastream content.

```
Connect = datastream('User1', 'Pass1', 'Datastream', ...  
    'http://dataworks.thomson.com/Dataworks/Enterprise/1.0')
```

See Also close, fetch, get, isconnection (Thomson Datastream functions)

Purpose

Request data from Thomson Datastream

Syntax

```
data = fetch(Connect, 'Security')
data = fetch(Connect, 'Security', 'Fields')
data = fetch(Connect, 'Security', 'Fields', 'Date')
data = fetch(Connect, Security, 'Fields', 'FromDate',
            'ToDate')
data = fetch(Connect, Security, 'Fields', 'FromDate',
            'ToDate', 'Period')
data = fetch(Connect, Security, 'Fields', 'FromDate',
            'ToDate', 'Period', 'Currency')
```

Arguments

Connect	Thomson Datastream connection object created with the <code>datastream</code> function.
Security	MATLAB string containing the name of a security in a format recognizable by the Thomson Datastream data server.
<i>Fields</i>	(Optional) MATLAB string or cell array of strings indicating the data fields for which data is to be retrieved.
Date	(Optional) MATLAB string indicating a specific calendar date for which data is requested.

FromDate (Optional) Start date for historical data.

Note Dates can be specified in any of the formats supported by `datestr` and `datenum` that show a year, month and day.

ToDate (Optional) End date for historical data. If *ToDate* is provided, *FromDate* cannot be an empty value.

Period (Optional) Period within a date range. *Period* values are
'd': daily values
'w': weekly values
'm': monthly values

Currency (Optional) Currency in which the data is reported.

Note You can input the optional arguments *Fields*, *FromDate*, *ToDate*, *Period*, and *Currency* as MATLAB strings or empty arrays (`[]`).

Description

`data = fetch(Connect, 'Security')` returns the default time series for the indicated security.

`data = fetch(Connect, 'Security', 'Fields')` returns data for the specified security and fields.

`data = fetch(Connect, 'Security', 'Fields', 'Date')` returns data for the specified security and fields on a particular date.

```
data = fetch(Connect, Security, 'Fields', 'FromDate',  
'ToDate') returns data for the specified security and fields for the  
indicated date range.
```

```
data = fetch(Connect, Security, 'Fields', 'FromDate',  
'ToDate', 'Period') returns instrument data for the given range  
with the indicated period.
```

```
data = fetch(Connect, Security, 'Fields', 'FromDate',  
'ToDate', 'Period', 'Currency') additionally specifies the currency  
in which the data is reported.
```

Examples

Here are some examples of using the Datastream fetch command to obtain data from Thomson Datastream.

Note The Datastream interface returns all data as strings. For example, when requesting Price data, it is returned in MATLAB as a cell array of strings within the structure. There is no way to determine the data type from the Datastream interface.

```
data = fetch(Connect, 'ICI') and data = fetch(Connect,  
'ICI', 'P') both return the trailing one-year price time series for the  
given instrument. ('P' is the default value for the 'Field' argument.)
```

```
data = fetch(Connect, 'ICI', {'P', 'PO'}, '08/01/2005')  
returns the closing and opening prices for the given instrument on the  
given date.
```

```
data = fetch(Connect, {'ICI', 'IBM'}, {'P', 'PO'},  
'8/01/2003', '8/01/2005', 'M') returns the monthly closing and  
opening prices for the indicated securities during the specified date  
range.
```

See Also

close, datastream, get, isconnection (Thomson Datastream functions)

get

Purpose Thomson Datastream connection properties

Syntax
`value = get(Connect, 'PropertyName')`
`value = get(Connect)`

Arguments

<code>Connect</code>	Thomson Datastream connection object created with the <code>datastream</code> function.
<code>PropertyName</code>	(Optional) A MATLAB string or cell array of strings containing property names. Property names are user datasource endpoint wsdl sources systeminfo version

Description `value = get(Connect, 'PropertyName')` returns the value of the specified properties for the Thomson Datastream connection object.
`value = get(Connect)` returns a MATLAB structure where each field name is the name of a property of `Connect`, and each field contains the value of that property.

See Also `close`, `datastream`, `fetch`, `isconnection` (Thomson Datastream functions)

Purpose	True if valid Thomson Datastream connection		
Syntax	<code>x = isconnection(Connect)</code>		
Arguments	<table><tr><td>Connect</td><td>Thomson Datastream connection object created with the datastream function.</td></tr></table>	Connect	Thomson Datastream connection object created with the datastream function.
Connect	Thomson Datastream connection object created with the datastream function.		
Description	<code>x = isconnection(Connect)</code> returns <code>x = 1</code> if the connection is a valid Thomson Datastream connection, and <code>x = 0</code> if it is not.		
Examples	<p>The function</p> <pre>c = datastream</pre> <p>establishes a connection to the Thomson Datastream API. Then<pre>x = isconnection(c) x = 1</pre><p>indicates that <code>c</code> is a valid Thomson Datastream connection.</p></p>		
See Also	<code>close</code> , <code>datastream</code> , <code>fetch</code> , <code>get</code> (Thomson Datastream functions)		

FactSet

close	Close FactSet connection
close	Close FactSet connection
factset	Connect to FactSet
factset	Connect to FactSet
fetch	Request data from FactSet
fetch	Request data from FactSet
get	FactSet connection properties
get	FactSet connection properties
isconnection	True if valid FactSet connection
isconnection	True if valid FactSet connection

Purpose Close FactSet connection

Syntax `close(Connect)`

Arguments

Connect	FactSet connection object created with the factset function.
---------	--

Description `close(Connect)` closes the connection to the FactSet data server.

See Also `factset`

factset

Purpose Connect to FactSet

Syntax `Connect = factset('UserName', 'SerialNumber', 'Password', 'ID')`

Arguments

UserName	User login name.
SerialNumber	User serial number.
Password	User password.
ID	FactSet customer identification number.

FactSet assigns the values for all of the above input arguments.

Description `Connect = factset('UserName', 'SerialNumber', 'Password', 'ID')` connects to the FactSet FAST interface.

Examples

```
Connect = factset('username', '1234', 'password', 'fsid')
Connect =

    user: 'username'
  serial: '1234'
password: 'password'
    cid: 'fsid'
```

See Also `close`, `fetch`, `get`, `isconnection` (FactSet functions)

Purpose

Request data from FactSet

Syntax

```
data = fetch(Connect)
data = fetch(Connect, 'Library')
data = fetch(Connect, 'Security', 'Fields')
data = fetch(Connect, 'Security', 'Fields', 'Date')
data = fetch(Connect, 'Security', 'Fields', 'FromDate',
            'ToDate')
data = fetch(Connect, 'Security', 'FromDate', 'ToDate',
            'Period')
```

Arguments

Connect	FactSet connection object created with the factset function.
Library	FactSet formula library.
Security	A MATLAB string or cell array of strings containing the names of securities in a format recognizable by the FactSet server.
<i>Fields</i>	A MATLAB string or cell array of strings indicating the data fields for which data is to be retrieved.
Date	Date string or serial date number indicating date for the requested data. If today's date is entered, yesterday's data is returned.
FromDate	Beginning date for date range.

Note Dates can be specified in any of the formats supported by `datestr` and `datenum` that show a year, month and day.

fetch

<code>ToDate</code>	End date for date range.
<code>Period</code>	Period within date range. <i>Period</i> values are 'd': daily values 'b': business day daily values 'm': monthly values 'mb': beginning monthly values 'me': ending monthly values 'q': quarterly values 'qb': beginning quarterly values 'qe': ending quarterly values 'y': annual values 'yb': beginning annual values 'ye': ending annual values

Description

`data = fetch(Connect)` returns the names of all available formula libraries.

`data = fetch(Connect, 'Library')` returns the valid field names for a given formula library.

`data = fetch(Connect, 'Security', 'Fields')` returns data for the specified security and fields.

`data = fetch(Connect, 'Security', 'Fields', 'Date')` returns security data for the specified fields on the requested date.

`data = fetch(Connect, 'Security', 'Fields', 'FromDate', 'ToDate')` returns security data for the specified fields for the date range `FromDate` to `ToDate`.

`data = fetch(Connect, 'Security', 'FromDate', 'ToDate', 'Period')` returns security data for the date range `FromDate` to `ToDate` with the indicated period.

Examples

Example 1: Obtain the names of the available formula libraries.

```
D = fetch(Connect)
```

Example 2: Obtain the valid field names for the FactSetSecurityCalcs library.

```
D = fetch(Connect, 'fs')
```

Example 3: Obtain closing price of a given security.

```
D = fetch(Connect, 'ABC', 'price')
```

Example 4: Obtain the closing price for the given dates for a given security using the default period of the data.

```
D = fetch(C, 'ABC', 'price', '8/01/99', '8/10/99')
```

Example 5: Obtain the monthly closing price for the given dates for a given security.

```
D = fetch(C, 'ABC', 'price', '8/01/99', '8/10/99', 'm')
```

See Also

close, factset, get, isconnection (FactSet functions)

get

Purpose FactSet connection properties

Syntax
`value = get(Connect, 'PropertyName')`
`value = get(Connect)`

Arguments

<code>Connect</code>	FactSet connection object created with the factset function.
<code>PropertyName</code>	(Optional) A MATLAB string or cell array of strings containing property names. Property names are user serial password cid

Description

`value = get(Connect, 'PropertyName')` returns the value of the specified properties for the FactSet connection object.

`value = get(Connect)` returns a MATLAB structure where each field name is the name of a property of Connect, and each field contains the value of that property.

Examples Use the factset function to establish a connection to FactSet.

```
Connect = factset('Fast_User', '1234', 'Fast_Pass', 'userid')
```


Now use the `get` function to retrieve the connection property value.

```
h = get(Connect)

h=
    user: 'Fast_User'
  serial: '1234'
password: 'Fast_Pass'
   cid: 'userid'

get(Connect, 'user')

ans =

Fast_User
```

See Also

`close`, `fetch`, `factset`, `isconnection` (FactSet functions)

isconnection

Purpose True if valid FactSet connection

Syntax `x = isconnection(Connect)`

Arguments

Connect FactSet connection object created with the factset function.

Description `x = isconnection(Connect)` returns `x = 1` if the connection is a valid FactSet connection, and `x = 0` if it is not.

Examples

The function

```
c = factset
```

establishes a FactSet connection.

Then

```
x = isconnection(c);
```

```
x = 1
```

indicates that `c` is a valid FactSet connection.

See Also

`close`, `fetch`, `factset`, `get` (FactSet functions)

Hyperfeed

close	Close Hyperfeed connection
close	Close Hyperfeed connection
fetch	Request data from Hyperfeed
fetch	Request data from Hyperfeed
get	Hyperfeed connection properties
get	Hyperfeed connection properties
hyperfeed	Connect to Hyperfeed
hyperfeed	Connect to Hyperfeed
isconnection	True if valid Hyperfeed connection
isconnection	True if valid Hyperfeed connection

close

Purpose	Close Hyperfeed connection		
Syntax	<code>close(Connect)</code>		
Arguments	<table><tr><td>Connect</td><td>Hyperfeed connection object created with the <code>hyperfeed</code> function.</td></tr></table>	Connect	Hyperfeed connection object created with the <code>hyperfeed</code> function.
Connect	Hyperfeed connection object created with the <code>hyperfeed</code> function.		
Description	<code>close(Connect)</code> closes the connection to the Hyperfeed data server.		
See Also	<code>hyperfeed</code>		

Purpose Request data from Hyperfeed

Syntax

```

data = fetch(Connect, 'Security')
data = fetch(Connect, 'Security', 'Fields')
data = fetch(Connect, 'Security', 'Date')
data = fetch(Connect, 'Security', 'Fields', 'Date')
data = fetch(Connect, 'Security', 'FromDate', 'ToDate')
data = fetch(Connect, 'Security', 'Fields', 'FromDate',
    'ToDate')
data = fetch(Connect, 'Security', 'FromDate', 'ToDate',
    'Period')

```

Arguments

Connect	Hyperfeed connection object created with the hyperfeed function.
Security	A MATLAB string or cell array of strings containing the names of a securities in a format recognizable by the Hyperfeed server.
<i>Fields</i>	A MATLAB string or cell array of strings indicating the data fields for which data is to be retrieved. Some possible values are Symbol Last Date Time Change Open High Low Volume
Date	Date string or serial date number indicating date for the requested data. If today's date is entered, yesterday's data is returned.

`FromDate` Beginning date for historical data.

Note Dates can be specified in any of the formats supported by `datestr` and `datenum` that show a year, month and day.

`ToDate` End date for historical data.

`Period` Period within date range. *Period* values are

'd': daily
'w': weekly
'm': monthly
'v': dividends

Description

`data = fetch(Connect, 'Security')` returns data for all fields from Hyperfeed's Web site for the indicated securities.

`data = fetch(Connect, 'Security', 'Fields')` returns data for the specified fields.

`data = fetch(Connect, 'Security', 'Date')` returns all security data for the requested date.

`data = fetch(Connect, 'Security', 'Fields', 'Date')` returns security data for the specified fields on the requested date.

`data = fetch(Connect, 'Security', 'FromDate', 'ToDate')` returns security data for the date range `FromDate` to `ToDate`.

`data = fetch(Connect, 'Security', 'Fields', 'FromDate', 'ToDate')` returns security data for the specified fields for the date range `FromDate` to `ToDate`.

`data = fetch(Connect, 'Security', 'FromDate', 'ToDate', 'Period')` returns security data for the date range `FromDate` to `ToDate` with the indicated period.

Examples

Obtain the closing price for Coca Cola on April 6, 2000.

```
c = hyperfeed('History');  
  
ClosePrice = fetch(c, 'ko', 'Close', 'Apr 6 00')  
  
ClosePrice =  
  
730582.00      45.75
```

See Also

close, get, hyperfeed, isconnection (Hyperfeed functions)

get

Purpose Hyperfeed connection properties

Syntax `value = get(Connect, 'PropertyName')`
`value = get(Connect)`

Arguments

<code>Connect</code>	Hyperfeed connection object created with the hyperfeed function.
<code>PropertyName</code>	(Optional) A MATLAB string or cell array of strings containing property names. Property names are Connection IPAddress Port Socket Version

Description `value = get(Connect, 'PropertyName')` returns the value of the specified properties for the Hyperfeed connection object.
`value = get(Connect)` returns a MATLAB structure where each field name is the name of a property of `Connect`, and each field contains the value of that property.

Examples Use the hyperfeed function to establish a connection to Hyperfeed.

```
c = hyperfeed('Price')
```

Now use the `get` function to retrieve the connection property value.

```
h = get(c, Connection)
```

```
h=
```

```
connection: 3  
table: 'Price'
```


See Also close, fetch, hyperfeed, isconnection (Hyperfeed functions)

hyperfeed

Purpose	Connect to Hyperfeed		
Syntax	Connect = hyperfeed(<i>Table</i>)		
Arguments	<table><tr><td><i>Table</i></td><td>A MATLAB string indicating the Hyperfeed table (database) to access. Possible values are Price (Default) Profile History</td></tr></table>	<i>Table</i>	A MATLAB string indicating the Hyperfeed table (database) to access. Possible values are Price (Default) Profile History
<i>Table</i>	A MATLAB string indicating the Hyperfeed table (database) to access. Possible values are Price (Default) Profile History		
Description	Connect = hyperfeed(<i>Table</i>) connects to the indicated Hyperfeed table.		
Examples	<pre>c = hyperfeed('Price')</pre> <p>connects to the Hyperfeed Price table.</p>		
See Also	close, fetch, get, isconnection (Hyperfeed functions)		

Purpose	True if valid Hyperfeed connection		
Syntax	<code>x = isconnection(Connect)</code>		
Arguments	<table><tr><td>Connect</td><td>Hyperfeed connection object created with the hyperfeed function.</td></tr></table>	Connect	Hyperfeed connection object created with the hyperfeed function.
Connect	Hyperfeed connection object created with the hyperfeed function.		
Description	<code>x = isconnection(Connect)</code> returns <code>x = 1</code> if the connection is a valid Hyperfeed connection, and <code>x = 0</code> if it is not.		
Examples	<p>The function</p> <pre>c = hyperfeed</pre> <p>establishes a Hyperfeed connection, <code>c</code>, to the Price table.</p> <p>Then</p> <pre>x = isconnection(c);</pre> <pre>x = 1</pre> <p>indicates that <code>c</code> is a valid Hyperfeed connection.</p>		
See Also	<code>close</code> , <code>fetch</code> , <code>get</code> , <code>hyperfeed</code> (Hyperfeed functions)		

FT Interactive Data

close	Close FT Interactive Data connection
close	Close FT Interactive Data connection
fetch	Request data from FT Interactive Data
fetch	Request data from FT Interactive Data
get	FT Interactive Data connection properties
get	FT Interactive Data connection properties
idc	Connect to FT Interactive Data
idc	Connect to FT Interactive Data
isconnection	True if valid FT Interactive Data connection
isconnection	True if valid FT Interactive Data connection

Purpose	Close FT Interactive Data connection		
Syntax	<code>close(Connect)</code>		
Arguments	<table><tr><td><code>Connect</code></td><td>FT Interactive Data connection object created with the <code>idc</code> function.</td></tr></table>	<code>Connect</code>	FT Interactive Data connection object created with the <code>idc</code> function.
<code>Connect</code>	FT Interactive Data connection object created with the <code>idc</code> function.		
Description	<code>close(Connect)</code> closes the connection to the FT Interactive Data server.		
Examples	<pre>c = idc establishes an FT Interactive Data connection, c. close(c) closes this connection.</pre>		
See Also	<code>idc</code>		

fetch

Purpose Request data from FT Interactive Data

Syntax

```
data = fetch(Connect, 'Security', 'Fields')
data = fetch(Connect, 'Security', 'Fields', 'FromDate',
            'ToDate')
data = fetch(Connect, 'Security', 'Fields', 'FromDate',
            'ToDate', 'Period')
data = fetch(Connect, '', 'GUILookup', 'GUICategory')
```

Arguments

Connect	FT Interactive Data connection object created with the <code>idc</code> function.
Security	A MATLAB string containing the name of a security in a format recognizable by the FT Interactive Data server.
<i>Fields</i>	A MATLAB string or cell array of strings indicating specific fields for which data is to be provided. Valid field names are in the file <code>@idc/idcfields.mat</code> . The variable <code>bbfieldnames</code> contains the list of field names.
FromDate	Beginning date for historical data.
<hr/> Note Dates can be specified in any of the formats supported by <code>datestr</code> and <code>datenum</code> that show a year, month and day. <hr/>	
ToDate	End date for historical data.
<i>Period</i>	Period within date range.
<i>GUICategory</i>	GUI category. Possible values are F (All valid field categories) S (All valid security categories)

Description

`data = fetch(Connect, 'Security', 'Fields')` returns data for the indicated fields of the designated securities. Load the file `idc/idcfields` to see the list of supported fields.

`data = fetch(Connect, 'Security', 'Fields', 'FromDate', 'ToDate')` returns historical data for the indicated fields of the designated securities.

`data = fetch(Connect, 'Security', 'Fields', 'FromDate', 'ToDate', 'Period')` returns historical data for the indicated fields of the designated securities with the designated dates and period. Consult the Remote Plus documentation for a list of valid *Period* values.

`data = fetch(Connect, '', 'GUILookup', 'GUICategory')` opens the FT Interactive Data dialog box for selecting fields or securities.

Examples

```
D = fetch(Connect, '', 'GUILOOKUP', 'S')
```

opens the dialog box for looking up securities.

```
D = fetch(Connect, '', 'GUILOOKUP', 'F')
```

opens the dialog box for selecting fields.

See Also

`close`, `get`, `idc`, `isconnection` (FT Interactive Data functions)

get

Purpose FT Interactive Data connection properties

Syntax `value = get(Connect, 'PropertyName')`
`value = get(Connect)`

Arguments

<code>Connect</code>	FT Interactive Data connection object created with the <code>idc</code> function.
<code>PropertyName</code>	(Optional) A MATLAB string or cell array of strings containing property names. Property names are <code>Connected</code> <code>Connection</code> <code>Queued</code>

Description `value = get(Connect, 'PropertyName')` returns the value of the specified properties for the FT Interactive Data connection object. `'PropertyName'` is a string or cell array of strings containing property names.

`value = get(Connect)` returns a MATLAB structure. Each field name is the name of a property of `Connect`, and each field contains the value of that property.

See Also `close`, `fetch`, `idc`, `isconnection` (FT Interactive Data functions)

Purpose	Connect to FT Interactive Data
Syntax	<code>Connect = idc</code>
Description	<code>Connect = idc</code> connects to the FT Interactive Data server. <code>Connect</code> is a connection handle used by other functions to obtain data.
Examples	<pre>c = idc</pre> <p>makes a connection to the FT Interactive Data server.</p>
See Also	<code>close</code> , <code>fetch</code> , <code>get</code> , <code>isconnection</code> (FT Interactive Data functions)

isconnection

Purpose True if valid FT Interactive Data connection

Syntax `x = isconnection(Connect)`

Arguments Connect FT Interactive Data connection object created with the `idc` function.

Description `x = isconnection(Connect)` returns `x = 1` if the connection is a valid FT Interactive Data connection, and `x = 0` if it is not.

Examples The function

```
c = idc
```

establishes an FT Interactive Data connection, `c`.

Then

```
x = isconnection(c)
x = 1
```

indicates that `c` is a valid FT Interactive Data connection.

See Also `close`, `fetch`, `get`, `idc` (FT Interactive Data functions)

Federal Reserve Economic Data

close	Close FRED connection
close	Close FRED connection
fetch	Request data from FRED
fetch	Request data from FRED
fred	Connect to FRED
fred	Connect to FRED
get	FRED connection properties
get	FRED connection properties
isconnection	True if valid FRED connection
isconnection	True if valid FRED connection

fred

Purpose Connect to FRED

Syntax
Connect = fred(URL)
Connect = fred

Arguments URL Create a connection using a specified URL.

Description Connect = fred(URL) establishes a connection to a FRED data server.
Connect = fred verifies that the URL
`http://research.stlouisfed.org/fred2/series` is accessible and
creates a connection.

Examples
`c = fred(http://research.stlouisfed.org/fred2/series)`
makes a connection to the FRED server at the specified URL.

See Also close, fetch, get, isconnection

Purpose Close FRED connection

Syntax `close(Connect)`

Arguments

<code>Connect</code>	FRED connection object created with the <code>fred</code> function.
----------------------	---

Description `close(Connect)` closes the connection to the FRED data server.

Examples

```
c = fred(http://research.stlouisfed.org/fred2/series)
```

establishes a FRED connection, `c`.

```
close(c)
```

closes this connection.

See Also `fred`

fetch

Purpose Request data from FRED

Syntax

```
data = fetch(Connect, 'Security')  
data = fetch(Connect, 'Security', 'D1')  
data = fetch(Connect, 'Security', 'D1', 'D2')
```

Arguments

Connect	FRED connection object created with the fred function.
Security	MATLAB string containing the name of a security in a format recognizable by the FRED server.
'D1'	MATLAB string or date number indicating the date from which data is to be retrieved. Returns all data for the date 'D1'; if 'D1' is today's date, the data from yesterday will be returned.
'D2'	MATLAB string or date number indicating the date range from which data is to be retrieved. Returns all data for the given security for the date range 'D1' to 'D2'.

Note You can specify dates in any of the formats supported by datestr and datenum that show a year, month, and day.

Description For a given security, fetch returns historical data using the FRED connection.

Examples Fetch all available daily U.S. to Euro foreign exchange rates:

```
d = fetch(f, 'DEXUSEU')
```

```

d =
    Title: ' U.S. / Euro Foreign Exchange Rate '
    SeriesID: ' DEXUSEU '
    Source: ' Board of Governors of the Federal Reserve System '
    Release: ' H.10 Foreign Exchange Rates '
    SeasonalAdjustment: ' Not Applicable '
    Frequency: ' Daily '
    Units: ' U.S. Dollars to One Euro '
    DateRange: ' 1999-01-04 to 2006-06-19 '
    LastUpdated: ' 2006-06-20 9:39 AM CT '
    Notes: ' Noon buying rates in New York City for cable transfers payable '
    Data: [1877x2 double]

```

Fetch data for a given date range:

```

d = fetch(f, 'DEXUSEU', '01/01/2006', '06/01/2006')
d =
    Title: ' U.S. / Euro Foreign Exchange Rate '
    SeriesID: ' DEXUSEU '
    Source: ' Board of Governors of the Federal Reserve System '
    Release: ' H.10 Foreign Exchange Rates '
    SeasonalAdjustment: ' Not Applicable '
    Frequency: ' Daily '
    Units: ' U.S. Dollars to One Euro '
    DateRange: ' 1999-01-04 to 2006-06-19 '
    LastUpdated: ' 2006-06-20 9:39 AM CT '
    Notes: ' Noon buying rates in New York City for cable transfers payable '
    Data: [105x2 double]

```

See Also

close, get, isconnection

get

Purpose FRED connection properties

Syntax

```
value = get(Connect, 'PropertyName')  
value = get(Connect)
```

Arguments

Connect	FRED connection object created with the fred function.
'PropertyName'	A MATLAB string or cell array of strings containing property names. Property names are: <ul style="list-style-type: none">• url• ip• port

Description

`value = get(Connect, 'PropertyName')` returns a MATLAB structure containing the value of the specified properties for the FRED connection object.

`value = get(Connect)` returns the value for all properties.

Examples

```
c = FRED(http://research.stlouisfed.org/fred2/series)
```

establishes a FRED connection, `c`.

The syntax

```
p = get(c, {'port', 'ip'})
```


returns

```
p =  
  port: 8194  
  ip: 111.222.33.444
```

See Also

close, fetch, isconnection

isconnection

Purpose True if valid FRED connection

Syntax `x = isconnection(Connect)`

Arguments

Connect FRED connection object created with the fred function.

Description `x = isconnection(Connect)` returns `x = 1` if the connection is a valid FRED connection, and `x = 0` if it is not.

Examples

The function

```
c = fred(http://research.stlouisfed.org/fred2/series)
```

establishes a FRED connection, `c`.

Then

```
x = isconnection(c)
x = 1
```

indicates that `c` is a valid FRED connection.

See Also

`close`, `fetch`, `get`

Yahoo!

close	Close Yahoo! connection
close	Close Yahoo! connection
fetch	Request data from Yahoo!
fetch	Request data from Yahoo!
get	Yahoo! connection properties
get	Yahoo! connection properties
isconnection	True if valid Yahoo! connection
isconnection	True if valid Yahoo! connection
yahoo	Connect to Yahoo!
yahoo	Connect to Yahoo!

close

Purpose	Close Yahoo! connection		
Syntax	<code>close(Connect)</code>		
Arguments	<table><tr><td><code>Connect</code></td><td>Yahoo! connection object created with the yahoo function.</td></tr></table>	<code>Connect</code>	Yahoo! connection object created with the yahoo function.
<code>Connect</code>	Yahoo! connection object created with the yahoo function.		
Description	<code>close(Connect)</code> closes the connection to the Yahoo! data server.		
See Also	yahoo		

Purpose

Request data from Yahoo!

Syntax

```
data = fetch(Connect, 'Security')
data = fetch(Connect, 'Security', 'Fields')
data = fetch(Connect, 'Security', 'Date')
data = fetch(Connect, 'Security', 'Fields', 'Date')
data = fetch(Connect, 'Security', 'FromDate', 'ToDate')
data = fetch(Connect, 'Security', 'Fields', 'FromDate',
    'ToDate')
data = fetch(Connect, 'Security', 'FromDate', 'ToDate',
    'Period')
```

Arguments

- Connect Yahoo! connection object created with the yahoo function.
- Security A MATLAB string or cell array of strings containing the name of a security in a format recognizable by the Yahoo! server.

Note Retrieving multiple securities at one time is not supported for Yahoo. You must fetch a single security at a time.

- Fields* A MATLAB string or cell array of strings indicating the data fields for which data is to be retrieved. A partial list of supported values for current market data are
 - Symbol
 - Last
 - Date
 - Time
 - Change
 - Open
 - High

fetch

	Low
	Volume
	A partial list of supported values for historical data are
	Close
	Date
	High
	Low
	Open
	Volume
	Adj. Close*
	For a complete list of supported values for market and historical data, see <code>yhfields.mat</code> .
Date	Date string or serial date number indicating date for the requested data. If today's date is entered, yesterday's data is returned.
FromDate	Beginning date for historical data.
<hr/> Note Dates can be specified in any of the formats supported by <code>datestr</code> and <code>datenum</code> that show a year, month and day. <hr/>	
ToDate	End date for historical data.
Period	Period within date range. <i>Period</i> values are 'd': daily 'w': weekly 'm': monthly 'v': dividends

Description

`data = fetch(Connect, 'Security')` returns data for all fields from Yahoo!'s Web site for the indicated security. Retrieving multiple

securities at one time is not supported for Yahoo. You must fetch a single security at a time

`data = fetch(Connect, 'Security', 'Fields')` returns data for the specified fields.

`data = fetch(Connect, 'Security', 'Date')` returns all security data for the requested date.

`data = fetch(Connect, 'Security', 'Fields', 'Date')` returns security data for the specified fields on the requested date.

`data = fetch(Connect, 'Security', 'FromDate', 'ToDate')` returns security data for the date range FromDate to ToDate.

`data = fetch(Connect, 'Security', 'Fields', 'FromDate', 'ToDate')` returns security data for the specified fields for the date range FromDate to ToDate.

`data = fetch(Connect, 'Security', 'FromDate', 'ToDate', 'Period')` returns security data for the date range FromDate to ToDate with the indicated period.

Examples

Example 1: Obtain the closing price for Coca Cola on April 6, 2000.

```
c = yahoo;
ClosePrice = fetch(c, 'ko', 'Close', 'Apr 6 00')
ClosePrice =
           730582.00           45.75
```

Example 2: Use the Yahoo! data server to obtain the last prices for a set of equities.

```
y = yahoo;
FastFood = fetch(y, {'ko', 'pep', 'mcd'}, 'Last')
```

fetch

```
FastFood =  
    Last: [3x1 double]
```

```
FastFood.Last
```

```
ans =
```

```
    42.96  
    45.71  
    23.70
```

See Also

close, get, isconnection, yahoo (Yahoo! functions)

Purpose

Yahoo! connection properties

Syntax

```
value = get(Connect, 'PropertyName')  
value = get(Connect)
```

Arguments

Connect

Yahoo! connection object created with the yahoo function.

PropertyName

(Optional) A MATLAB string or cell array of strings containing property names. Currently the only property name recognized is url.

Description

`value = get(Connect, 'PropertyName')` returns the value of the specified properties for the Yahoo! connection object.

`value = get(Connect)` returns a MATLAB structure where each field name is the name of a property of Connect, and each field contains the value of that property.

Examples

Use the yahoo function to establish a connection to Yahoo!.

```
c = yahoo
```

```
c =
```

```
    url: 'http://quote.yahoo.com'
```

Now use the get function to retrieve the connection property value.

```
get(c, 'url')
```

```
ans =
```

```
    url: 'http://quote.yahoo.com'
```

See Also

close, fetch, isconnection, yahoo (Yahoo! functions)

isconnection

Purpose True if valid Yahoo! connection

Syntax `x = isconnection(Connect)`

Arguments Connect Yahoo! connection object created with the yahoo function.

Description `x = isconnection(Connect)` returns `x = 1` if the connection is a valid Yahoo! connection, and `x = 0` if it is not.

Examples The function

```
c = yahoo
```

establishes a Yahoo! connection, c.

Then

```
x = isconnection(c)
```

```
x = 1
```

indicates that c is a valid Yahoo! connection.

See Also `close`, `fetch`, `get`, `yahoo` (Yahoo! functions)

Purpose

Connect to Yahoo!

Syntax

```
Connect = yahoo
Connect = yahoo('URL', 'IPAddress', PortNumber)
```

Arguments

URL	Must be 'http://quote.yahoo.com'.
IPAddress	A MATLAB string containing the Internet address of proxy server machine.
PortNumber	Port number on proxy server.

Description

Connect = yahoo verifies that the URL http://quote.yahoo.com is accessible and creates a connection handle.

Connect = yahoo('URL', 'IPAddress', PortNumber) connects to Yahoo! through a proxy server using the IP address and port number provided. This form of the yahoo function may be required when connecting to Yahoo! from behind an internal firewall.

Examples

Use the yahoo function to establish a connection to the Yahoo! data server.

```
Connect = yahoo

Connect =

    url: 'http://quote.yahoo.com'
```

Use the yahoo function to establish a connection to the Yahoo! data server, providing an IP address and port number on a proxy server.

```
Connect = yahoo('http://quote.yahoo.com', '111.222.33.444', 5678)

Connect =

    url: 'http://quote.yahoo.com'
```

```
ip: '111.222.33.444'  
port: 5678
```

See Also close, fetch, get, isconnection (Yahoo! functions)

Related Information

Additional Software (p. A-2)

Obtaining client software and Web
connection

Additional Software

Obtaining Client Software

If you want to use the Datafeed Toolbox to retrieve data from Bloomberg, Hyperfeed, or FT Interactive Data Corporation data servers, you need to install client software available from each of these companies. If client software is not properly licensed for your machine, you will receive the error message

```
Invalid MEX-file
```

when attempting to connect to the data server.

Information about the services offered by these companies is available on the Web at

<http://www.bloomberg.com>

<http://www.hyperfeed.com>

<http://www.FTInteractiveData.com>

Contact your data server sales representative for information.

Connecting to FactSet

To connect to FactSet using the Datafeed Toolbox, you must be licensed to use FactSet's FAST technology. You can find further information on FactSet's Web site (<http://www.factset.com>).

Connecting to the Thomson Datastream API

To connect to the Thomson Datastream API via the Web, you require a user name, password, and URL, all of which Thomson will provide. For further information, consult Thomson's Web site (<http://www.thomson.com>).

B

- bloomberg 2-3 3-3
- Bloomberg
 - connection handle 1-4
 - connection object 1-4

C

- close 1-6
 - Bloomberg 2-4 3-4
 - FactSet 2-31 3-31
 - FRED 2-57 to 2-58 3-57 to 3-58
 - FT Interactive Data 2-49 3-49
 - Hyperfeed 2-40 3-40
 - Thomson Datastream 2-23 3-23
 - Yahoo! 2-64 3-64
- connecting 1-4
- connection handle 1-4
- connection object 1-4
- Connection tab 1-15
- CUSIP number 2-5 3-5

D

- data
 - default 1-7
 - field 1-9
 - header 1-7
 - historical 1-12
 - time series 1-11
- Data tab 1-18
- Datafeed dialog box 1-15
- datastream 2-24 3-24
- default data 1-7
- dftool 1-15
- disconnecting 1-6

F

- Federal Reserve Economic Data (FRED) 2-56 3-56
- fetch 1-7
 - Bloomberg 2-5 3-5
 - FactSet 2-33 3-33
 - FT Interactive Data 2-50 3-50
 - Hyperfeed 2-41 3-41
 - Thomson Datastream 2-25 3-25
 - Yahoo! 2-65 3-65
- field data 1-9
- field names 1-10
- Flag values 1-8

G

- get 1-6
 - Bloomberg 2-13 3-13
 - FactSet 2-36 3-36
 - FRED 2-60 3-60
 - FT Interactive Data 2-52 3-52
 - Hyperfeed 2-44 3-44
 - Thomson Datastream 2-28 3-28
 - Yahoo! 2-69 3-69
- GETDATA argument 1-9
- graphical user interface 1-15

H

- HEADER argument 1-8
- header data 1-7
- header fields 1-7
- historical data 1-12
- HISTORY argument 1-12
- hyperfeed 2-46 3-46

I

- idc 2-53 3-53
- IP address 1-4

isconnection 1-5
 Bloomberg 2-15 3-15
 FactSet 2-38 3-38
 FRED 2-62 3-62
 FT Interactive Data 2-54 3-54
 Hyperfeed 2-47 3-47
 Thomson Datastream 2-29 3-29
 Yahoo! 2-70 3-70

L

LOOKUP argument 1-13

M

markets 1-13

P

port number 1-4
pricevol 2-16 3-16

R

retrieve properties 1-5

S

Securities Lookup dialog box 1-21
showtrades 2-18 3-18
stockticker 2-20 3-20

T

ticker symbols 1-13
time series data 1-11
TIMESERIES argument 1-11

V

verification 1-5

Y

yahoo 2-71 3-71