Datafeed Toolbox

For Use with MATLAB®

- Computation
- Visualization
- Programming

User's Guide

Version 1

How to Contact The MathWorks:

`www.mathworks.com`	Web
`comp.soft-sys.matlab`	Newsgroup

`support@mathworks.com`	Technical support
`suggest@mathworks.com`	Product enhancement suggestions
`bugs@mathworks.com`	Bug reports
`doc@mathworks.com`	Documentation error reports
`service@mathworks.com`	Order status, license renewals, passcodes
`info@mathworks.com`	Sales, pricing, and general information

508-647-7000 Phone

508-647-7001 Fax

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

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

Datafeed Toolbox User's Guide

Printing 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+)

Contents

1 Getting Started

2 Function Reference

A Related Information

Index

1

Getting Started

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 Time Series Toolbox, for further analysis.

Communicating with a Financial Data Server

The Datafeed Toolbox supports connections to five financial data servers:

- Bloomberg (`http://www.bloomberg.com`)
- FactSet (`http://www.factset.com`)
- Hyperfeed (`http://www.hyperfeed.com/f_main.html`)
- FT Interactive Data Corporation (`http://www.FTInteractiveData.com`)
- Yahoo (`http://www.yahoo.com`)

Bloomberg, Hyperfeed, and FT Interactive Data all require that you install proprietary software on your PC. To connect to FactSet or Yahoo, you need to have access to the Internet.

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

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

This document uses the Bloomberg financial data server as an example of establishing communication and retrieving data. Other data servers work similarly.

Communication Management

For each of the supported financial data servers, the Datafeed Toolbox uses four commands to manage communication:

- `bloomberg`, `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 command

```
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` command. For a connection object `c` previously created with one of the above connection commands,

```
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 command `get`. This command 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` command 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 command

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

which returns

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

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

```
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` command with the format

```
close(Connect)
```

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

Retrieving Data

The `fetch` command 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` command to retrieve data from a Bloomberg data server. Versions of the `fetch` command 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 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` command.
- `Security` is the list of securities for which data is requested.
- 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` command 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 command

```
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 command with the syntax

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

For example, use the bloomberg command 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 command 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 command

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

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

```
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` command 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 1-9 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` command 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 returns

```
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 dialogs, 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, enables you to 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 1-14
- "Data Retrieval" on page 1-16

To learn about setting overrides on retrieved data, see

- "Setting Overrides" on page 1-18

For additional information about the **Securities Lookup** dialog box, see

- "Securities Lookup Dialog Box (Bloomberg, FT Interactive Data)" on page 1-19

Datafeed Dialog Box

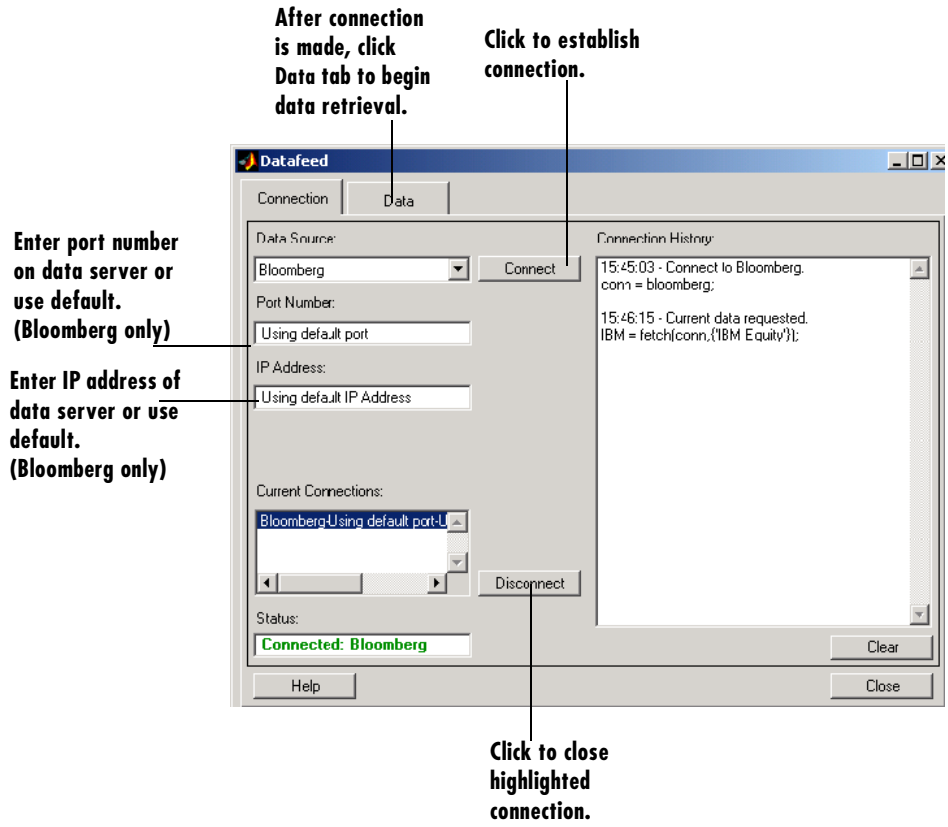
The **Datafeed** dialog box establishes the connection with the data server and manages the retrieval of data. Enter the command `dftool` to display the **Datafeed** dialog box on your screen. The **Datafeed** dialog box consists of two tabbed dialogs:

- The **Connection** tab establishes communication with a data server. (See "Connecting to a Data Server" on page 1-14.)
- The **Data** tab specifies the data request. (See "Data Retrieval" on page 1-16.)

Connecting to a Data Server

The **Connection** tab establishes a connection to one or more data servers. For FactSet, Yahoo and FT Interactive Data connections, choose the data server from the **Data Source** choices and click on 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 on the **Connect** button and

accept the default values provided when the Bloomberg software was installed on your machine.



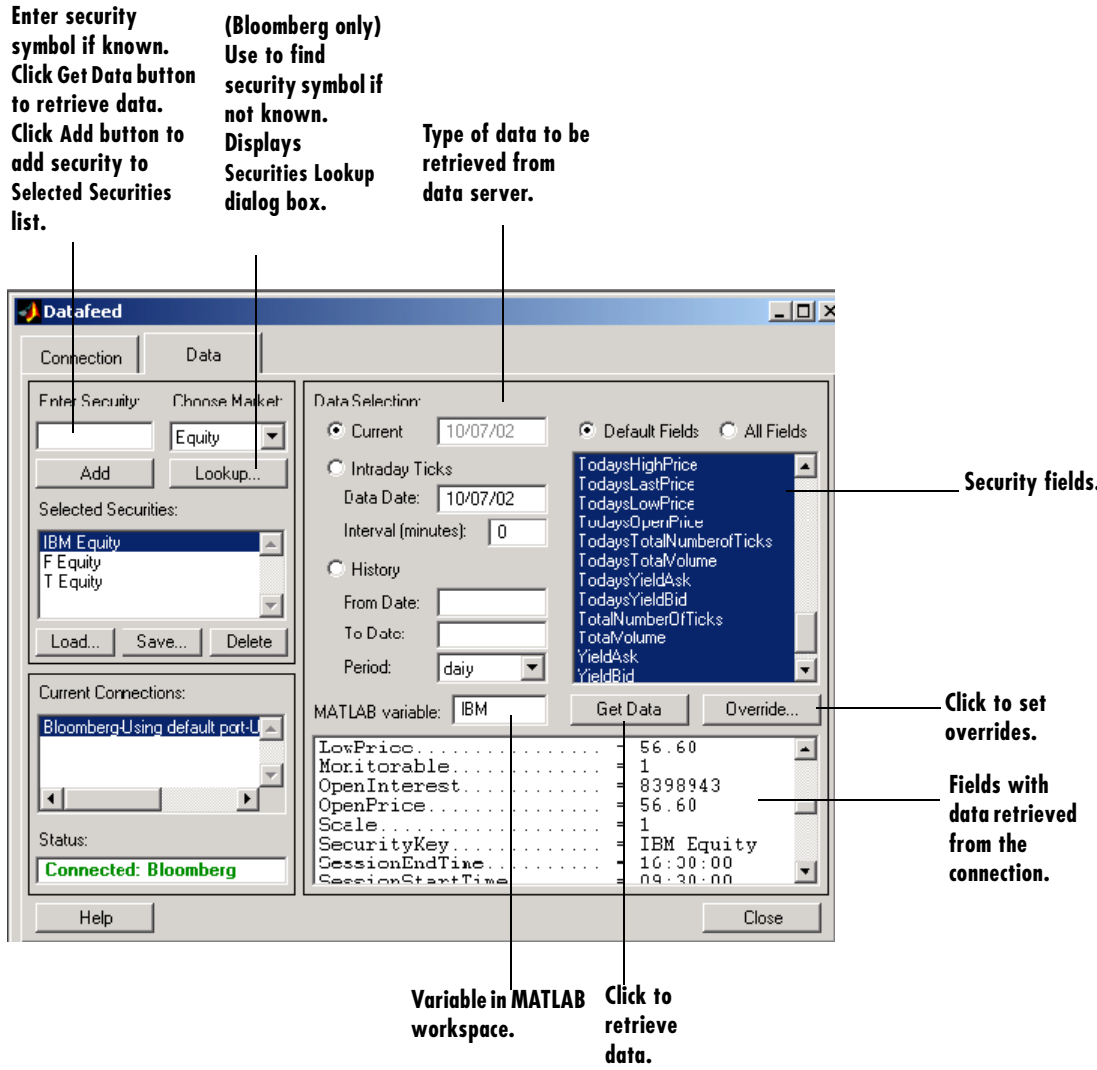
1. (Bloomberg only) Enter the port number on the data server in the **Port Number** box (or use default).
2. (Bloomberg only) Enter the IP address of the data server in the **IP Address** box (or use default).
3. Click the **Connect** button to establish the connection.

4 When the `Connected` message appears in the **Status** box, click on 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 1-16.

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.



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

2 Indicate the type of data you are seeking in the **Data Selection** panel.

3 Indicate whether you want the default or full set of data in the **Fields** panel.

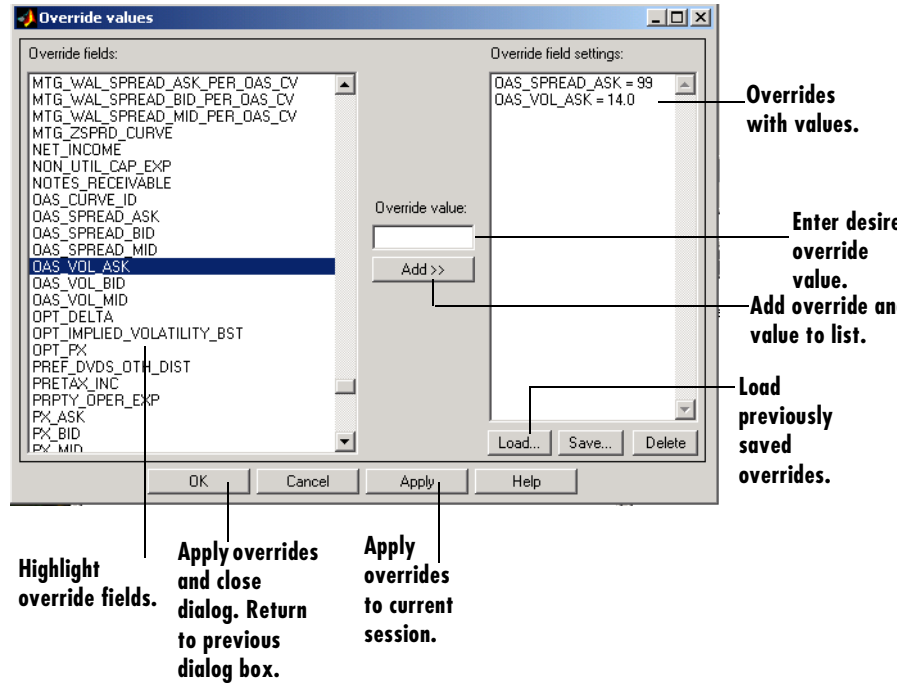
4 Click the **Get Data** button to retrieve data from the data server.

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 1-19.)

Setting Overrides

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

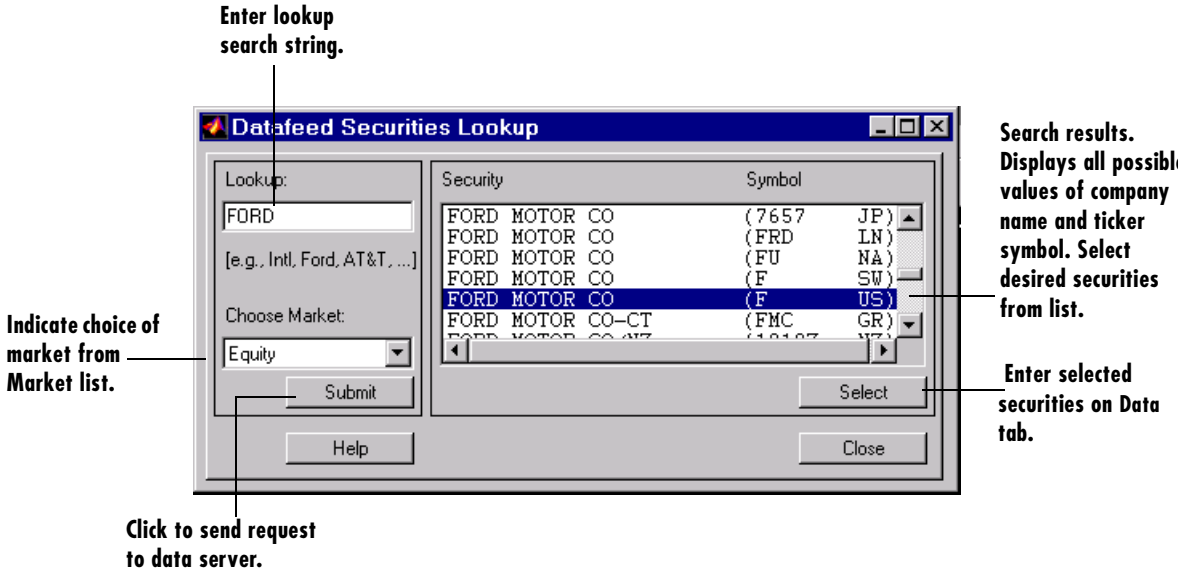


Securities Lookup Dialog Box (Bloomberg, FT Interactive Data)

Click on the **Lookup** button of the **Datafeed** dialog box **Data** tab to display the **Securities Lookup** dialog box. See “Data Retrieval” on page 1-16 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.


Enter lookup search string.
Datafeed Securities Lookup
Lookup:
FORD
[e.g., Intl, Ford, AT&T, ...]
Choose Market:
Equity
Submit
Help
Security
Symbol
FORD MOTOR CO (7657 JP)
FORD MOTOR CO (FRD LN)
FORD MOTOR CO (FU NA)
FORD MOTOR CO (F SW)
FORD MOTOR CO (F US)
FORD MOTOR CO-CT (FMC GR)
Select
Close
Search results. Displays all possible values of company name and ticker symbol. Select desired securities from list.
Indicate choice of market from Market list.
Enter selected securities on Data tab.
Click to send request to data server.

2

Function Reference

Functions - Categorical List

Datafeed Toolbox functions are categorized by the data service that provides financial data to MATLAB.

Bloomberg Functions

This section provides detailed descriptions of the Bloomberg functions in the Datafeed Toolbox.

`bloomberg`	Connect to Bloomberg
`close`	Close connection
`fetch`	Request data
`get`	Get connection properties
`isconnection`	True if valid connection

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`

close

Purpose

Close Bloomberg connection

Syntax

```
close(Connect)
```

Arguments

`Connect`	Bloomberg connection object created with the `bloomberg` command.

Description

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

Examples

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

establishes a Bloomberg connection, `c`.

```
close(c)
```

closes this connection.

See Also

`bloomberg`

Purpose

Request data from Bloomberg

Syntax

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

Arguments

`Connect`	Bloomberg connection object created with the `bloomberg` command.
`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: For **`HEADER`**, **`GETDATA`**, and **`MONITOR`** data only, `Security` may be a cell array of strings containing a list of securities.)
`Flag`	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 `Flag` value is not specified, `'DEFAULT'` is assumed. `TODAY`: Today's data only. `ENHANCED`: Data from most recent date of each individual field.

Fields	A MATLAB string or cell array of strings indicating specific fields for which data is requested. Valid field names are in the file `@bloomberg/bbfields.mat`. The variable `bbfieldnames` contains the list of field names.
`Override`	(Optional) String or cell array of strings containing override field list.
`Values`	(Optional) String or cell array of strings containing override field values.
`Date`	Date string or serial date number indicating date for the time series. Specify `now` for today's time series data.
`Minutes`	(Optional) Tick interval in minutes.
`TickField`	(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 command `dftool('ticktypes')` to return the list of intraday tick fields.
`FromDate`	Beginning date for historical data.
`ToDate`	End date for historical data.
Period	(Optional) Period of the data: `'d'` - daily, `'w'` - weekly, `'m'` - monthly, `'q'` - quarterly, `'y'` - yearly. If *Period* is not specified, the default period for the data is used.

Market	A MATLAB string indicating the market in which a particular security trades. *Market* values 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)
MATLABProg	A string that is the name of any valid MATLAB program written to parse the data structure created by a '**HEADER**' call.
NumTicks	(Optional) Number of ticks processed before '**MONITOR**' loop terminates.

Description

For a given security, `fetch` returns header (default), field, time series, 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`*`')` returns data 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')` returns the data for the specified fields of the indicated security. You can further specify the data with the optional `Override` and `Values` arguments.

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

- 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', 'Field', 'FromDate', ToDate)` 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.

`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.

`data = fetch(Connect, 'Security', 'MONITOR', 'MATLABProg', NumTicks)` subscribes to a specific security and peforms a '**HEADER**' call to obtain Bloomberg data. It then runs a MATLAB program that can parse the data structure created by the '**HEADER**' call. You terminate the '**MONITOR**' loop by typing **Ctrl+C**.

If you include the optional argument `NumTicks`, the '**MONITOR**' loop automatically terminates after `NumTicks` ticks have been processed. Enter **Ctrl+C** to terminate processing prior to `NumTicks` ticks having been reached.

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.

Monthly Closing Price

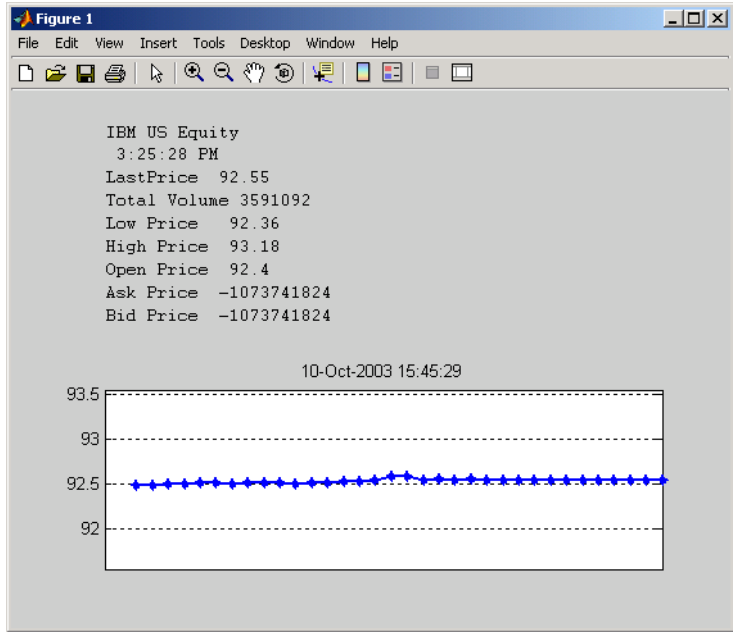
```
D = fetch(C,'ABC US Equity','HISTORY','Last_Price','8/01/99',...
'9/30/00','m')
```

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

Realtime Security Monitoring

```
D = fetch(C,'IBM US Equity','MONITOR', 'monitor1')
D = fetch(C,'IBM US Equity','MONITOR', 'monitor2', 1000)
```

subscribes to the security IBM and runs the monitoring function each time Bloomberg provides a tick signal. In the first example the monitoring runs until you type **Ctrl+C**. A sample of the graphic output that `monitor1` produces is shown below.



In the second example `monitor2` runs until you type **Ctrl+C** or until you have received 1000 ticks. The output of `monitor2` is in text format only.

The demonstration monitoring functions `monitor1` and `monitor2` are included in the Datafeed Toolbox.

See Also

`bloomberg`, `close`, `get`, `isconnection`

Purpose

Get Bloomberg connection properties

Syntax

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

Arguments

`Connect`	Bloomberg connection object created with the `bloomberg` command.

`PropertyName`	(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 command

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

returns

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

See Also

`bloomberg`, `close`, `fetch`, `isconnection`

isconnection

Purpose

True if valid Bloomberg connection

Syntax

```
x = isconnection(Connect)
```

Arguments

`Connect`	Bloomberg connection object created with the `bloomberg` command.

Description

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

Examples

The command

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

establishes a Bloomberg connection, `c`.

Then

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

indicates that `c` is a valid Bloomberg connection.

See Also

`bloomberg`, `close`, `fetch`, `get`

FactSet Functions

This section provides detailed descriptions of the FactSet functions in the Datafeed Toolbox.

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

close

Purpose

Close FactSet connection

Syntax

```
close(Connect)
```

Arguments

`Connect`	FactSet connection object created with the `factset` command.

Description

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

See Also

`factset`

Purpose

Connect to FactSet

Syntax

```
Connect = factset('UserName', 'SerialNumber', 'Password',
   'CustomerID')
```

Arguments

`UserName`	User login name
`SerialNumber`	User serial number
`Password`	User password
`CustomerID`	FactSet customer identification number

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

Description

`Connect = factset('UserName', 'SerialNumber', 'Password', 'CustomerID')` connects to the FactSet FastFetch 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)

fetch

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', 'Fields','FromDate', 'ToDate',
'Period')
```

Arguments

`Connect`	FactSet connection object created with the `factset` command.
`Library`	FactSet formula library.
`Security`	A MATLAB string or cell array of strings containing the names of a securities in a format recognizable by the FactSet server.
`Fields`	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.

`ToDate`	End date for date range.
`Period`	Period within date range. *`Period`* 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:

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

returns the names of the available formula libraries.

Example 2:

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

returns the valid field names for the `FactSetSecurityCalcs` library.

Example 3:

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

returns closing price of the given security.

Example 4:

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

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

Example 5:

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

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

See Also

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

Purpose

Get FactSet connection properties

Syntax

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

Arguments

`Connect`	FactSet connection object created with the `factset` command.

`PropertyName`	(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` command to establish a connection to FactSet.

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

Now use the `get` command 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)

Purpose

True if valid FactSet connection

Syntax

```
x = isconnection(Connect)
```

Arguments

`Connect`	FactSet connection object created with the `factset` command.

Description

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

Examples

The command

```
c = factset
```

establishes a FactSet connection.

Then

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

```
x = 1
```

indicates that `c` is a valid FactSet connection.

See Also

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

Hyperfeed Functions

This section provides detailed descriptions of the Hyperfeed functions in the Datafeed Toolbox.

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

Purpose

Close Hyperfeed connection

Syntax

```
close(Connect)
```

Arguments

`Connect`	Hyperfeed connection object created with the `hyperfeed` command.

Description

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

See Also

`hyperfeed`

fetch

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` command.
`Security`	A MATLAB string or cell array of strings containing the names of a securities in a format recognizable by the Hyperfeed server.

`Fields`	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.

`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)

Purpose

Get Hyperfeed connection properties

Syntax

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

Arguments

`Connect`	Hyperfeed connection object created with the `hyperfeed` command.

`PropertyName`	(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` command to establish a connection to Hyperfeed.

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

Now use the `get` command 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(Table)
```

Arguments

Table	Indicates the Hyperfeed table (database) to access. Possible values are:
	`'Price'` (Default)
	`'Profile'`
	`'History'`

Description

`Connect = hyperfeed(Table)` connects to the indicated Hyperfeed table.

Examples

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

connects to the Hyperfeed Price table.

See Also

`close`, `fetch`, `get`, `isconnection` (Hyperfeed functions)

Purpose

True if valid Hyperfeed connection

Syntax

```
x = isconnection(Connect)
```

Arguments

`Connect`	Hyperfeed connection object created with the `hyperfeed` command.

Description

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

Examples

The command

```
c = hyperfeed
```

establishes a Hyperfeed connection, `c`, to the Price table.

Then

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

```
x = 1
```

indicates that `c` is a valid Hyperfeed connection.

See Also

`close`, `fetch`, `get`, `hyperfeed` (Hyperfeed functions)

FT Interactive Data Functions

This section provides detailed descriptions of the FT Interactive Data functions in the Datafeed Toolbox.

`close`	Close connection
`fetch`	Request data
`get`	Get connection properties
`idc`	Connect to IDC
`isconnection`	True if valid connection

Purpose

Close FT Interactive Data connection

Syntax

```
close(Connect)
```

Arguments

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

Description

`close(Connect)` closes the connection to the FT Interactive Data server.

Examples

```
c = idc
```

establishes an FT Interactive Data connection, `c`.

```
close(c)
```

closes this connection.

See Also

`idc`

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, 'String', 'Lookup', 'Type', 'Market',
   NumRecords, StartRecord)
data = fetch(Connect,'','Lookup','Category')
data = fetch(Connect,'','GUILookup','GUICategory')
```

Arguments

`Connect`	FT Interactive Data connection object created with the `idc` command.
`Security`	A MATLAB string containing the name of a security in a format recognizable by the FT Interactive Data server.
`Fields`	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 `@idc/idcfields.mat`. The variable `bbfieldnames` contains the list of field names.
`FromDate`	Beginning date for historical data.
`ToDate`	End date for historical data.
`Period`	Period within date range.
`String`	Search string.
`Type`	Lookup type. Possible values are: `F` (Fields) `S` (Securities)
`Market`	Market to search.
`NumRecords`	Number of record to fetch.
`StartRecord`	Starting record for fetch.

Category	Lookup category. Possible values are: F (All valid field categories) S (All valid security categories)
GUICategory	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.

`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 period.

`data = fetch(Connect, 'String', '**Lookup**', '*Type*', '*Market*', NumRecords, StartRecord)` returns data of the requested type by searching within the designated market for the string.

`data = fetch(Connect,'','**Lookup**','*Category*')` returns all valid field or security categories.

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

Examples

```
d = fetch(c,'ford','lookup','s','equity',4,1)
```

returns the first four securities containing the string `'ford'` starting with the first record.

See Also

`close`, `get`, `idc`, `isconnection`

get

Purpose

Get FT Interactive Data connection properties

Syntax

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

Arguments

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

`PropertyName`	(optional) A MATLAB string or cell array of strings containing property names. Property names are:
	`Connected`
	`Connection`
	`Queued`

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

```
Connect = idc
```

Description

`Connect = idc` connects to the FT Interactive Data Corporation data server. `Connect` is a connection handle used by other functions to obtain data.

Examples

```
c = idc
```

makes a connection to the FT Interactive Data server.

See Also

`close`, `fetch`, `get`, `isconnection` (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` command.

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 command

```
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)

Yahoo Functions

This section provides detailed descriptions of the Yahoo functions in the Datafeed Toolbox.

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

close

Purpose

Close Yahoo connection

Syntax

```
close(Connect)
```

Arguments

`Connect`	Yahoo connection object created with the `yahoo` command.

Description

`close(Connect)` 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` command.
`Security`	A MATLAB string or cell array of strings containing the names of a securities in a format recognizable by the Yahoo server.

`Fields`	A MATLAB string or cell array of strings indicating the data fields for which data is to be retrieved. For current market data the values are:
	`Symbol`
	`Last`
	`Date`
	`Time`
	`Change`
	`Open`
	`High`
	`Low`
	`Volume`
	
	For historical data the values are:
	`Close`
	`Date`
	`High`
	`Low`
	`Open`
	`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.
`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 Yahoo'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

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
```

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')

FastFood =
    Last: [3x1 double]

FastFood.Last

ans =

         42.96
         45.71
         23.70
```

See Also

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

get

Purpose

Get Yahoo connection properties

Syntax

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

Arguments

`Connect`	Yahoo connection object created with the `yahoo` command.
`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` command to establish a connection to Yahoo.

```
c = yahoo

c =

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

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

```
get(c,  url )

ans =

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

See Also

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

Purpose

True if valid Yahoo connection

Syntax

```
x = isconnection(Connect)
```

Arguments

`Connect`	Yahoo connection object created with the `yahoo` command.

Description

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

Examples

The command

```
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)

yahoo

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` command may be required when connecting to Yahoo from behind an internal firewall.

Examples

Use the `yahoo` command to establish a connection to the Yahoo data server.

```
Connect = yahoo

Connect =

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

Use the `yahoo` command 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)

A

Related Information

Additional 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/f_main.html`

`http://www.FTInteractiveData.com`

Contact your data server sales representative for information.

Installation and Configuration

To install the Datafeed Toolbox, see the MATLAB Installation documentation for your computer system.

For information about installing Bloomberg, Hyperfeed, or FT Interactive Data Corporation software on your system, contact your sales representative from these companies.

Index