# Datafeed Toolbox

**For Use with MATLAB®**

**Computation**

**Visualization**

**Programming**

The MATH WORKS Inc.

User's Guide

*Version 1.2*

**How to Contact The MathWorks:**

508-647-7000 Phone

508-647-7001 Fax

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

| | |
|---|---|
| `http://www.mathworks.com` | Web |
| `ftp.mathworks.com` | Anonymous FTP server |
| `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 |
| `subscribe@mathworks.com` | Subscribing user registration |
| `service@mathworks.com` | Order status, license renewals, passcodes |
| `info@mathworks.com` | Sales, pricing, and general information |

*Datafeed Toolbox User's Guide*

© COPYRIGHT 1999 - 2000 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 or for the federal government of the United States. By accepting delivery of the Program, the government hereby agrees that this software qualifies as "commercial" computer software within the meaning of FAR Part 12.212, DFARS Part 227.7202-1, DFARS Part 227.7202-3, DFARS Part 252.227-7013, and DFARS Part 252.227-7014. The terms and conditions of The MathWorks, Inc. Software License Agreement shall pertain to the government's use and disclosure of the Program and Documentation, and shall supersede any conflicting contractual terms or conditions. If this license fails to meet the government's minimum needs or is inconsistent in any respect with federal procurement law, the government agrees to return the Program and Documentation, unused, to MathWorks.

MATLAB, Simulink, Stateflow, Handle Graphics, and Real-Time Workshop are registered trademarks, and Target Language Compiler is a trademark of The MathWorks, Inc.

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

Printing History: December 1999 First printing New for MATLAB 5.3 (Release 11)
June 2000 Revised for Version 1.2 online only

# Contents

## Preface



## 1 Tutorial

## 2 Reference



## A Directory Structure

# Preface

# Using this Guide

This book describes the Datafeed Toolbox for MATLAB®. The Datafeed Toolbox enables MATLAB users to obtain financial data from a financial data server and to enter this data into the MATLAB environment.

## Organization of the Document

| Chapter | Description |
|---|---|
| Chapter 1. "Tutorial" | Describes how to connect to a data server and request various forms of data. |
| Chapter 2. "Reference" | Describes the specific functions used to connect to a data server and to request various forms of data. |
| Appendix A. "Directory Structure" | Shows the location of directories and files created after installation of the Datafeed Toolbox on your system. |

## Typographical Conventions

We use some or all of these conventions in our manuals.

| Item | Convention to Use | Example |
|---|---|---|
| Example code | `Monospace` font | To assign the value 5 to `A`, enter<br>`A = 5` |
| Function names/syntax | `Monospace` font | The `cos` function finds the cosine of each array element.<br>Syntax line example is<br>`MLGetVar ML_var_name` |
| Keys | **Boldface** with an initial capital letter | Press the **Return** key. |

| Item | Convention to Use | Example |
|---|---|---|
| Literal string (in syntax descriptions in Reference chapters) | `Monospace bold` for literals. | `f = freqspace(n,'whole')` |
| Mathematical expressions | Variables in *italics*<br>Functions, operators, and constants in standard text. | This vector represents the polynomial<br>$p = x^2 + 2x + 3$ |
| MATLAB output | `Monospace` font | MATLAB responds with<br>`A =`<br>`5` |
| Menu names, menu items, and controls | **Boldface** with an initial capital letter | Choose the **File** menu. |
| New terms | *Italics* | An *array* is an ordered collection of information. |
| String variables (from a finite list) | *Monospace italics* | `sysc = d2c(sysd, 'method')` |

# R12 Related Products

The MathWorks provides several products that are especially relevant to the kinds of tasks you can perform with the Datafeed Toolbox.

For more information about any of these products, see either:

- The online documentation for that product, if it is installed or if you are reading the documentation from the CD
- The MathWorks Web site, at `http://www.mathworks.com`; see the "products" section

---

**Note** The toolboxes listed below all include functions that extend MATLAB's capabilities.

---

**Table 0-1:**

| Product | Description |
|---|---|
| Database Toolbox | Tool for connecting to, and interacting with, most ODBC/JDBC databases from within MATLAB |
| Excel Link | Tool that integrates MATLAB capabilities with Microsoft Excel for Windows |
| Financial Derivatives Toolbox | Tool that extends the Financial Toolbox in the areas of fixed income derivatives and of securities contingent to interest rates, with functions for analyzing individual financial derivative instruments and portfolios composed of them |
| Financial Time Series Toolbox | Tool for analyzing time series data in the financial markets |

**Table 0-1:**

| Product | Description |
|---|---|
| Financial Toolbox | MATLAB functions for quantitative financial modeling and analytic prototyping |
| GARCH Toolbox | MATLAB functions for univariate Generalized Autoregressive Conditional Heteroskedasticity (GARCH) volatility modeling |
| Optimization Toolbox | Tool for general and large-scale optimization of nonlinear problems, as well as for linear programming, quadratic programming, nonlinear least squares, and solving nonlinear equations |
| Statistics Toolbox | Tool for analyzing historical data, modeling systems, developing statistical algorithms, and learning and teaching statistics |

## Additional Software

If you want to use the Datafeed Toolbox to retrieve data from Bloomberg or Interactive Data Corporation (IDC) data servers, you need to install client software available from each of these companies. Contact your sales representative for information. Information about the services offered by these companies is available on the Web at `http://www.bloomberg.com` and `http://www.intdata.com`.

## Installation and Configuration

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

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

1

# Tutorial

# Introduction

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 three financial data servers:

- Bloomberg (`http://www.bloomberg.com`)
- Interactive Data Corporation (IDC) (`http://www.idc.com`)
- Yahoo (`http://www.yahoo.com`)

Bloomberg and IDC both require that you install proprietary software on your PC. To connect to Yahoo, you need to have access to the Internet and to install a Web browser. The most commonly used browsers, Microsoft Internet Explorer (`http://www.microsoft.com`) and Netscape Communicator (`http://www.netscape.com`) are available for free download.

## Communication Management

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

- `bloomberg`, `idc`, `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 similiarly.

### 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
              OpenPrice: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 complete list of valid field names is stored in the file `@bloomberg/bbfields.mat`. Use the command

```
load @bloomberg/bbfields
```

to load this file. The variable `bbfieldnames` contains the list of field names. This list includes the header field names plus numerous others.

**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', {'OpenPrice';'LastPrice'})
```

returns

```
d =
    OpenPrice: 126.2500
    LastPrice: 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 forms of the command

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

or

```
data = fetch(Connection, Security, 'TICKS').
```

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

```
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 1-8 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(connection, 'IBM US Equity', 'HISTORY',...
'LastPrice', '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 is the date represented as a MATLAB date number, and column 2 is 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 1-13
- "Data Retrieval" on page 1-16

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

- "Securities Lookup Dialog Box (Bloomberg only)" on page 1-17

## Datafeed Dialog Box

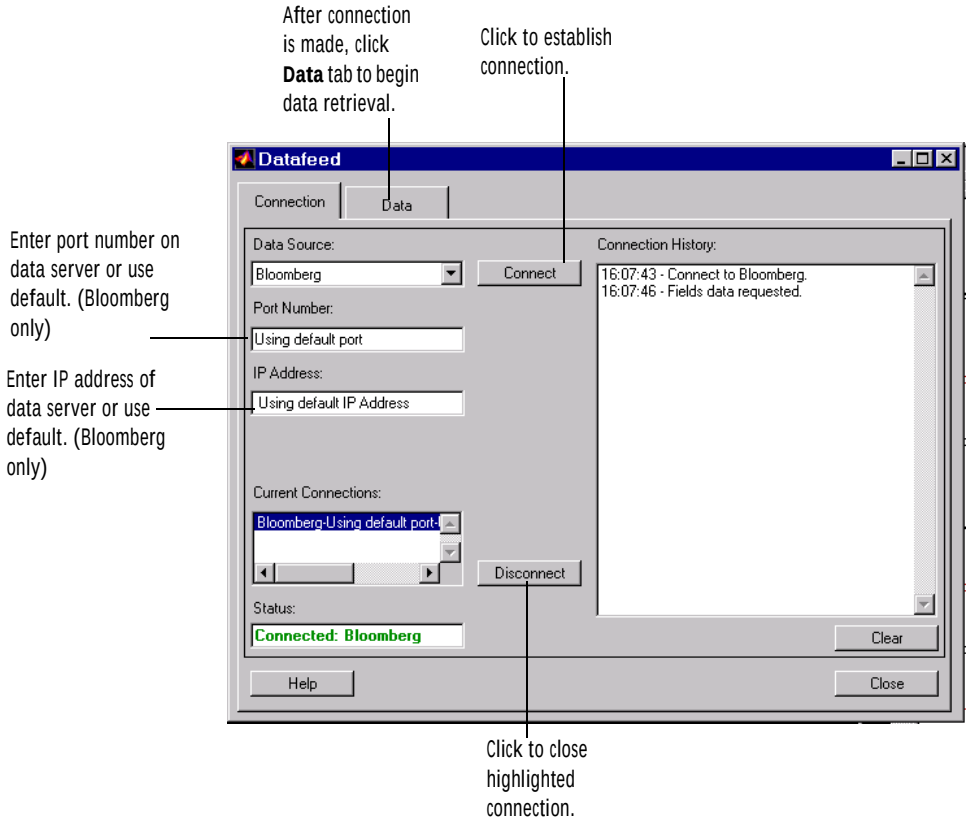
The **Datafeed** dialog box establishes the connection with the data server and manages the retrieval of data. Enter the command `dftool` to display the **Datafeed** dialog box on your screen. The **Datafeed** dialog box consists of two tabbed dialogs:

- The **Connection** tab establishes communication with a data server. (See "Connecting to a Data Server" on page 1-13.)
- 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 Yahoo and IDC 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).

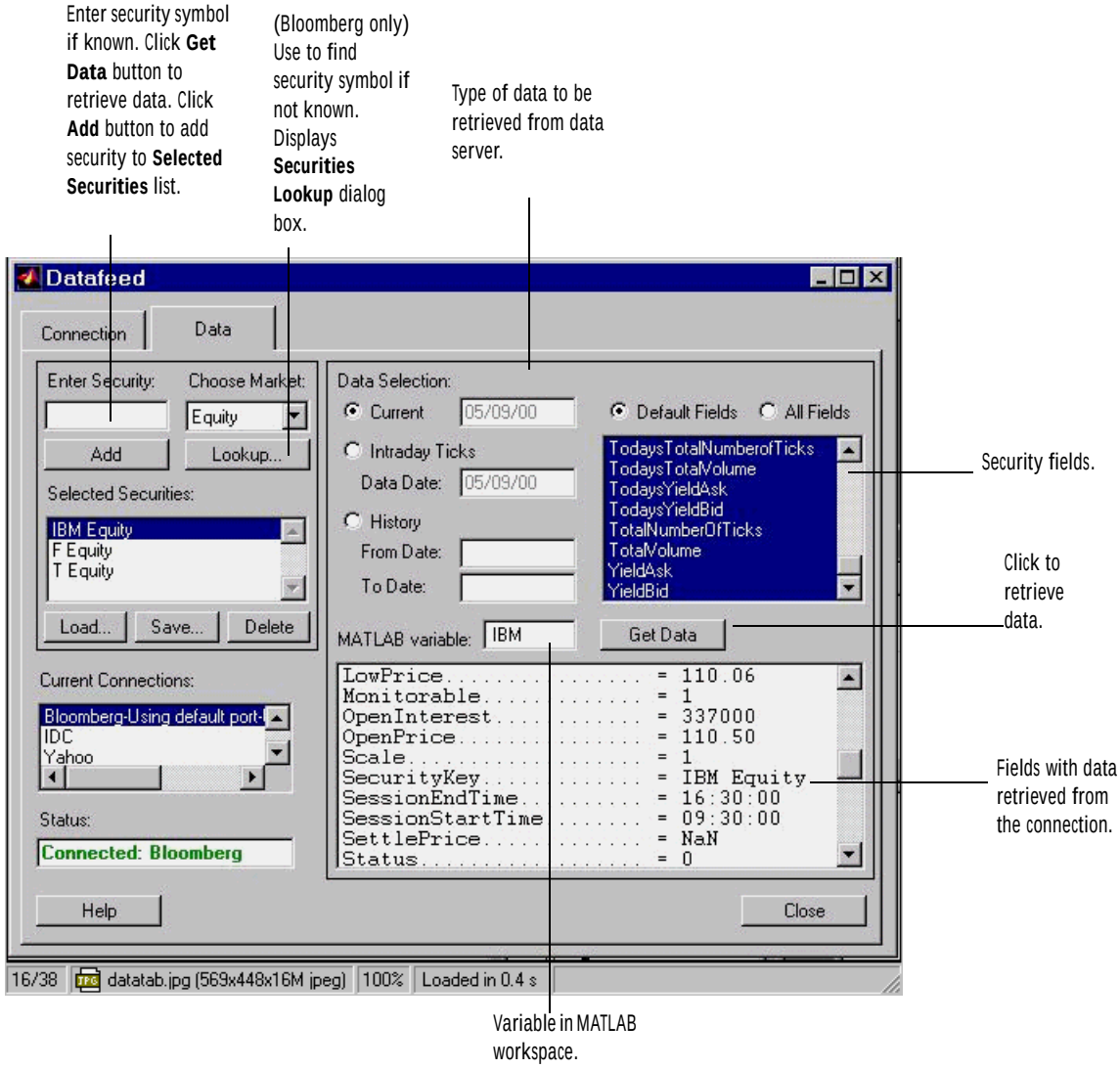
3. Click the **Connect** button to establish the connection.

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.



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.

---

**Bloomberg Users** 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 only)" on page 1-17.)

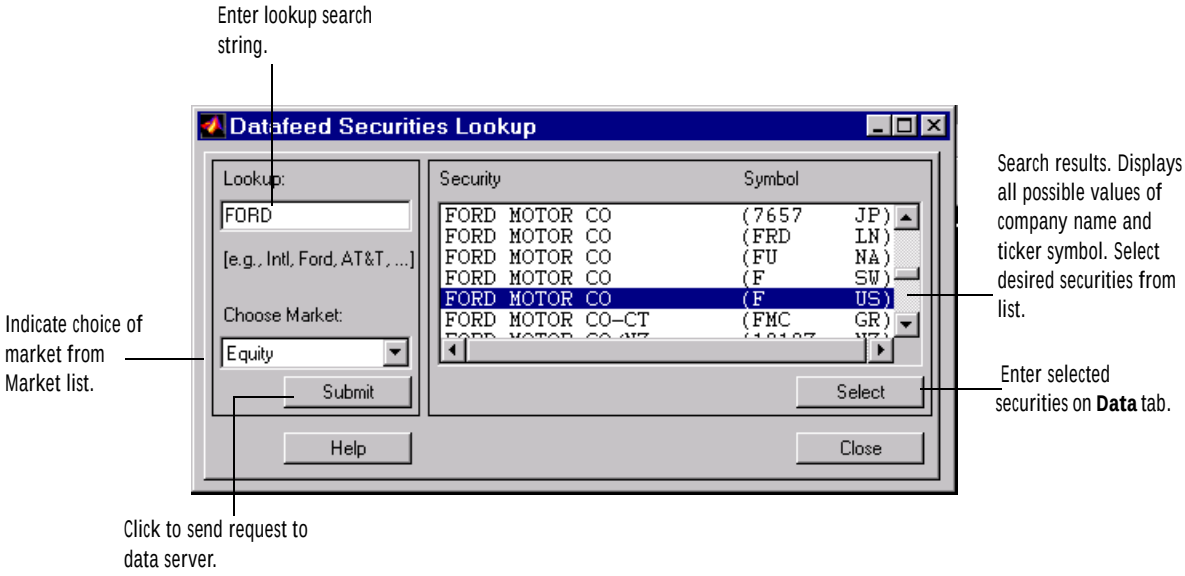
---

## Securities Lookup Dialog Box (Bloomberg only)

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 onthe **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 securites 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

# Reference

# Bloomberg Function Summary

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

**Table 2-1: Datafeed Toolbox Functions**

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

# bloomberg

**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.

**Example**

```
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`

**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.

**Example**

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

establishes a Bloomberg connection, `c`.

```
close(c)
```

closes this connection.

**See Also**

`bloomberg`

# fetch

## 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', 'TICKS')
data = fetch(Connect, 'Security', 'TIMESERIES', 'Date')
data = fetch(Connect, 'Security', 'HISTORY', 'Field', 'FromDate',
   'ToDate')
ticker = fetch(Connect, 'SearchString', 'LOOKUP', 'Market')
```

## 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. (Note: For header 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:<br><br>`DEFAULT`: Data from most recent bid, ask, or trade. If a `Flag` value is not specified, `'DEFAULT'` is assumed.<br><br>`TODAY`: Today's data only.<br><br>`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 to be provided. Valid field names are in the file `@bloomberg/bbfields.mat`. The variable `bbfieldnames` contains the list of field names. |
| `Date` | Date string or serial date number indicating date for the time series. Specify `now` for today's time series data. |
| `FromDate` | Beginning date for historical data. |

| | |
|---|---|
| `ToDate` | End date for historical data. |
| *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) |

## 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.

`data = fetch(Connect, 'Security', 'TICKS')` returns today's tick data.

`data = fetch(Connect, 'Security', 'TIMESERIES', 'Date')` returns the tick data for a security for the specified date. For today's tick data, specify

```
data = fetch(Connect, 'Security', 'TIMESERIES', now)
data = fetch(Connect, 'Security', 'TICKS') is an equivalent command.
```

`data = fetch(Connect, 'Security', 'HISTORY', 'Field', 'FromDate', ToDate)` returns historical data for the specified field for the date range `FromDate` to `ToDate`.

`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.

## Examples

For a United States equity with ticker ABC:

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

returns the header data.

```
D = fetch(C,'ABC US Equity','GETDATA',{'LastPrice';'OpenPrice'})
```

returns the opening and closing prices.

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

return today's time series.

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

returns the closing price for the given dates.

## 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:<br><br>`Connection`<br>`IPAddress`<br>`Port`<br>`Socket`<br>`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.

## Example

```
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.

**Example**

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`

# IDC Function Summary

This chapter provides detailed descriptions of the IDC functions in the Datafeed Toolbox.

**Table 2-2: Datafeed Toolbox IDC Functions**

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

# close

**Purpose** Close IDC connection

**Syntax** `close(Connect)`

**Arguments** `Connect` IDC connection object created with the `idc` command.

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

**Example**

```
c = idc
```

establishes an IDC connection, `c`.

```
close(c)
```

closes this connection.

**See Also** `idc`

## Purpose

Request data from IDC

## 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` | IDC connection object created with the `idc` command. |
| `Security` | A MATLAB string containing the name of a security in a format recognizable by the IDC 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:<br>`F` (Fields)<br>`S` (Securities) |
| *`Market`* | Market to search. |
| `NumRecords` | Number of record to fetch. |
| `StartRecord` | Starting record for fetch. |

| | |
|---|---|
| *Category* | Lookup category. Possible values are:<br>F (All valid field categories)<br>S (All valid security categories) |
| *GUICategory* | GUI category. Possible values are:<br>F (All valid field categories)<br>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 IDC 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`

## Purpose

Get IDC connection properties

## Syntax

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

## Arguments

| | |
|---|---|
| `Connect` | IDC connection object created with the `idc` command. |
| *`PropertyName`* | (optional) A MATLAB string or cell array of strings containing property names. Property names are:<br><br>`Connected`<br>`Connection`<br>`Queued` |

## Description

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

## Example

## See Also

`close`, `fetch`, `idc`, `isconnection` (IDC functions)

# idc

**Purpose**

Connect to IDC

**Syntax**

```
Connect = idc
```

**Description**

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

**Example**

```
c = idc
```

makes a connection to the IDC server.

**See Also**

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

**Purpose**

True if valid IDC connection

**Syntax**

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

**Arguments**

| | |
|---|---|
| `Connect` | IDC connection object created with the `idc` command. |

**Description**

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

**Example**

The command

```
c = idc
```

establishes an IDC connection, `c`.

Then

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

indicates that `c` is a valid IDC connection.

**See Also**

`close`, `fetch`, `get`, `idc` (IDC functions)

# Yahoo Function Summary

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

**Table 2-3: Datafeed Toolbox Yahoo Functions**

| Function | Purpose |
|---|---|
| `close` | Close connection |
| `fetch` | Request data |
| `get` | Get connection properties |
| `isconnection` | True if valid connection |
| `yahoo` | Connect to Yahoo. |

**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`

# fetch

**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. Possible values are:<br><br>`Symbol`<br>`Last`<br>`Date`<br>`Time`<br>`Change`<br>`Open`<br>`High`<br>`Low`<br>`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:<br>`d` (daily)<br>`w` (weekly)<br>`m` (monthly)<br>`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

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
```

**See Also** `close`, `get`, `isconnection`, `yahoo` (Yahoo functions)

## 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.

## Example

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)

# isconnection

**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.

**Example**

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)

**Purpose**

Connect to Yahoo

**Syntax**

```
Connect = yahoo
```

**Description**

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

**Example**

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

```
Connect = yahoo

Connect =

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

**See Also**

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

A

# Directory Structure

MATLAB is distributed in compressed format on CD-ROM. The installation procedure moves the files to your hard disk, decompresses them, and installs them into your MATLAB root directory. After installation of the MATLAB Datafeed Toolbox, your MATLAB directory should include these additional files and subdirectories.

---

**Note** In the directory structure shown below, the notation `<matlab>` represents your MATLAB root directory, the location where MATLAB is installed on your system.

---

**Table A-1: <matlab>/datafeed/datafeed/@bloomberg**

| File | Purpose |
|---|---|
| `bloomberg` | Connect to Bloomberg data server |
| `close` | Close connection to Bloomberg data server |
| `fetch` | Request data from Bloomberg data server |
| `get` | Get Bloomberg connection properties |
| `isconnection` | True if valid Bloomberg connection |

**Table A-2: <matlab>/datafeed/datafeed/@idc**

| File | Purpose |
|---|---|
| `close` | Close connection to IDC data server |
| `fetch` | Request data from IDC data server |
| `get` | Get IDC connection properties |
| `idc` | Connect to IDC data server |
| `isconnection` | True if valid IDC connection |

**Table A-3: <matlab>/datafeed/datafeed/@yahoo**

| File | Purpose |
|---|---|
| `close` | Close connection to Yahoo data server |
| `fetch` | Request data from Yahoo data server |
| `get` | Get Yahoo connection properties |
| `isconnection` | True if valid Yahoo connection |
| `yahoo` | Connect to Yahoo data server |

**Table A-4: <matlab>/datafeed/dfgui**

| File | Purpose |
|---|---|
| `dftool` | Datafeed graphical user interface |

# Index