TFX: An R interface to the TrueFX(tm) Web API

Garrett See

The **TFX** package is available on CRAN.

```
# CRAN version
install.packages("TFX")
# development version
#install.packages("TFX", repos="http://r-forge.r-project.org")
```

Unauthenticated Session

The **TFX** package is a simple interface to the TrueFX(tm) Web API. If you call `QueryTrueFX()` with its defaults, it will create an unauthenticated session and return quotes for 10 currency pairs. (All timestamps are in GMT.)

```
library(TFX)
QueryTrueFX()
```

```
##     Symbol Bid.Price Ask.Price      High       Low               TimeStamp
## 1  EUR/USD   1.29923   1.29928   1.29987   1.29838 2012-12-02 23:31:27.750
## 2  USD/JPY  82.48900  82.49300  82.50500  82.35300 2012-12-02 23:31:25.118
## 3  GBP/USD   1.60211   1.60218   1.60281   1.60112 2012-12-02 23:31:25.924
## 4  EUR/GBP   0.81091   0.81101   0.81168   0.81035 2012-12-02 23:31:26.415
## 5  USD/CHF   0.92766   0.92779   0.92843   0.92698 2012-12-02 23:31:27.505
## 6  EUR/JPY 107.17100 107.18100 107.18900 106.91200 2012-12-02 23:31:25.851
## 7  EUR/CHF   1.20521   1.20541   1.20569   1.20472 2012-12-02 23:31:25.492
## 8  USD/CAD   0.99407   0.99420   0.99461   0.99297 2012-12-02 23:31:17.497
## 9  AUD/USD   1.04231   1.04239   1.04341   1.04193 2012-12-02 23:31:27.355
## 10 GBP/JPY 132.15600 132.16900 132.20100 131.88700 2012-12-02 23:31:27.522
```

Note: if you don't see millisecond resolution, set `options(digits.secs=3)` or higher

Authenticated Session

Benefits of an Authenticated session

- Can request only the pairs that you want.
- Quotes are available for more pairs: at the time of this writing, there are 26 pairs available, but see the TrueFX Market Data Web API Developer Guide for the current list.
- Allows you to choose to only receive updates for pairs that have changed since the last request
- Results can be returned in different formats (html or csv). I'm not sure how much of a benefit this is (see the Formats and Parsing section)

In order to make customized requests, you must create an authenticated session which requires a username and password. Go to TrueFX.com to register for a free TrueFX(tm) account; you only need to provide an e-mail address.

I'll pretend that you chose "JSTrader" as your username and "Ou812" for your password because that is the example used in the official TrueFX Market Data Web API Developer Guide, but you should use your actual, authenticated username and password instead.

Get data for specific currency pairs

Once you have a registered username and password, you can create an authenticated session. With an authenticated session, you can request only the pairs that you want.

```
yen <- ConnectTrueFX("USD/JPY,EUR/JPY,GBP/JPY,AUD/JPY,CAD/JPY,CHF/JPY",
                     username="JSTrader", password="Ou812")
QueryTrueFX(yen)
```

```
##    Symbol Bid.Price Ask.Price    High     Low               TimeStamp
## 1 USD/JPY    82.489    82.493  82.505  82.353 2012-12-02 23:31:29.311
## 2 EUR/JPY   107.171   107.181 107.189 106.912 2012-12-02 23:31:25.851
## 3 GBP/JPY   132.156   132.169 132.201 131.887 2012-12-02 23:31:27.522
## 4 AUD/JPY    85.980    85.990  86.011  85.871 2012-12-02 23:31:26.509
## 5 CAD/JPY    82.971    82.984  82.996  82.825 2012-12-02 23:31:25.336
## 6 CHF/JPY    88.914    88.927  88.945  88.700 2012-12-02 23:31:28.328
```

If you pass an empty string (`""`) to the `currency.pairs` argument, the 15 pairs for which TrueFX(tm) provides historical data will be used (for a full list of

`currency.pairs` available for real-time quotes see the TrueFX Market Data Web API Developer Guide which you can view in your pdf viewer by calling `TrueFXRef()`)

```
QueryTrueFX(ConnectTrueFX("", "JSTrader", "Ou812", snapshot=TRUE))
```

```
##     Symbol Bid.Price Ask.Price      High       Low               TimeStamp
## 1  EUR/USD   1.29923   1.29928   1.29987   1.29838 2012-12-02 23:31:27.750
## 2  USD/JPY  82.48900  82.49300  82.50500  82.35300 2012-12-02 23:31:29.536
## 3  GBP/USD   1.60211   1.60218   1.60281   1.60112 2012-12-02 23:31:25.924
## 4  EUR/GBP   0.81091   0.81101   0.81168   0.81035 2012-12-02 23:31:26.415
## 5  USD/CHF   0.92766   0.92779   0.92843   0.92698 2012-12-02 23:31:27.505
## 6  EUR/JPY 107.17400 107.18100 107.18900 106.91200 2012-12-02 23:31:29.526
## 7  EUR/CHF   1.20521   1.20541   1.20569   1.20472 2012-12-02 23:31:25.492
## 8  USD/CAD   0.99407   0.99420   0.99461   0.99297 2012-12-02 23:31:17.497
## 9  AUD/USD   1.04231   1.04239   1.04341   1.04193 2012-12-02 23:31:27.355
## 10 GBP/JPY 132.15600 132.16900 132.20100 131.88700 2012-12-02 23:31:29.569
## 11 AUD/JPY  85.98000  85.99000  86.01100  85.87100 2012-12-02 23:31:26.509
## 12 AUD/NZD   1.27285   1.27321   1.27334   1.27022 2012-12-02 23:31:27.812
## 13 CAD/JPY  82.97100  82.98400  82.99600  82.82500 2012-12-02 23:31:25.336
## 14 CHF/JPY  88.91400  88.92700  88.94500  88.70000 2012-12-02 23:31:28.328
## 15 NZD/USD   0.81867   0.81887   0.82089   0.81852 2012-12-02 23:31:29.427
```

Retrieve only incremental updates

Authenticated sessions retrieve only incremental updates by default. This can be easily demonstrated by creating a session with a lot of symbols and making multiple requests. You'll see that an authenticated session does not always return data for all pairs by default.

If `ConnectTrueFX` is called with `snapshot=TRUE` (as above), then the session that is created will be disconnected after a request and will have to be reconnected for each request. By default, the `QueryTrueFX` function, will automatically reconnect a disconnected session but can be called with `reconnect=FALSE` if you want to throw an error when there is an attempt to request data with a disconnected session. Also note that even non-snapshot sessions become disconnected after roughly a minute of inactivity.

```
# this session gets only incremental updates
up <- ConnectTrueFX("", "JSTrader", "Ou812", snapshot=FALSE)
# this one is disconnected after a request and will have to be reconnected
# for a subsequent request.
ss <- ConnectTrueFX("", "JSTrader", "Ou812", snapshot=TRUE)
```

Now, make multiple requests and look at how many pairs are updated in each call.

the `up` session was created with `snapshot=FALSE` so it only gets updates for pairs that have values that have changed since the last request.

```
# make 20 requests and count the rows of each response
sapply(1:20, function(i) nrow(QueryTrueFX(up)))
```

```
##  [1] 15  5  3  1  1  2  0  2  4  0  0  0  2  1  1  1  0  0  1  0
```

The `ss` connection was created with `snapshot=TRUE`, so each time it is passed to `QueryTrueFX` it gets temporarily reconnected.

```
sapply(1:20, function(i) nrow(QueryTrueFX(ss)))
```

```
##  [1] 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15
```

Note that the snapshot session is no longer active.

```
isActive(ss)
```

```
## [1] FALSE
```

But the other one is still connected

```
isActive(up)
```

```
## [1] TRUE
```

The `reconnect` argument to `QueryTrueFX()` Let's make a new snapshot session that doesn't have as many pairs just so that the output won't take up as much space

```
eurss <- ConnectTrueFX("EUR/USD", "JSTrader", "Ou812", snapshot = TRUE)
```

The first request with a snapshot session returns data and then the session is immediately disconnected.

```
QueryTrueFX(eurss, reconnect = FALSE)  #after the request, eurss will be disconnected
```

```
##    Symbol Bid.Price Ask.Price    High     Low               TimeStamp
## 1 EUR/USD    1.2992   1.29927 1.29987 1.29838 2012-12-02 23:31:32.802
```

```
QueryTrueFX(eurss, reconnect = FALSE)  # Error; no longer connected
```

```
## Error: 'session' is not connected and 'reconnect' is not TRUE
```

```
Reconnect(eurss)
QueryTrueFX(eurss, reconnect = FALSE)  #disconnected after request
```

```
##    Symbol Bid.Price Ask.Price    High     Low               TimeStamp
## 1 EUR/USD   1.29919   1.29927 1.29987 1.29838 2012-12-02 23:31:35.642
```

If QueryTrueFX is called with reconnect=TRUE (the default) it will do the re-connection step for you.

```
isActive(eurss)
```

```
## [1] FALSE
```

```
QueryTrueFX(eurss)
```

```
##    Symbol Bid.Price Ask.Price    High     Low               TimeStamp
## 1 EUR/USD   1.29919   1.29927 1.29987 1.29838 2012-12-02 23:31:35.642
```

Other ways a session becomes disconnected. A snapshot session terminates after it has been used. There are two other ways for a session to become disconnected (regardless of whether it is a snapshot session or not).

1. Users can manually disconnect a session with `Disconnect()`
2. Sessions time out after roughly one minute and become disconnected.*

** technically, sessions timeout in about 70 seconds, but, since this is not documented in the Developer Guide, `isActive` returns `FALSE` after 60 seconds of inactivity to ensure no false positives*

```
isActive(up)
```

```
## [1] TRUE
```

```
Disconnect(up)
isActive(up)
```

```
## [1] FALSE
```

```
Reconnect(up)
isActive(up)
```

```
## [1] TRUE
```

```
#Sys.sleep(70) # session will become inactive after roughly 60 seconds
#isActive(up) # FALSE
```

Formats and Parsing

ParseTrueFX

By default, `QueryTrueFX()` will parse the results of a query (by calling `ParseTrueFX()`) and return a `data.frame`. You can choose not to parse the response by using `parse.response=FALSE`.

```
QueryTrueFX(ConnectTrueFX('USD/CAD,EUR/JPY', 'JSTrader', 'Ou812'),
            parse.response=FALSE)
```

```
## [1] "USD/CADEUR/JPY0.99107.4071710.99107.4201810.99461107.1890.99297106.9121354491077498
```

(`ParseTrueFX()` can be used to make sense of that string)

The `format` argument to `ConnectTrueFX()` Above you see what TrueFX(tm) returns when you use the `"default"` format. My advice is to always use the `"default"` format, because, last I checked, the `"default"` format gets updates before the other formats do. The next quickest to update is the `"html"` format, and `"csv"` is the slowest to update. The one benefit to using `"html"` or `"csv"` is that in addition to the columns returned when format is `"default"`, they also return an `Open` column. However, in addition to being delayed relative to the `"default"` format, the values for High and Low are backwards (i.e. **wrong**) for these formats.

For the sake of completeness, let's look at them anyway. I'll use `parse.response=FALSE` to show the differences in the raw responses. If you use `parse.response=TRUE` (the default) the response will be passed through `ParseTrueFX()` to get a `data.frame` by default, or a `list` if `pretty=FALSE`.

```
QueryTrueFX(ConnectTrueFX("USD/CAD,EUR/JPY", 'JSTrader', 'Ou812', format='html'),
            parse.response=FALSE)
```

```
## [1] "<table><tr><td>USD/CAD</td><td>1354491069334</td><td>0.99</td><td>407</td><td>0.99</
```

```
QueryTrueFX(ConnectTrueFX("USD/CAD,EUR/JPY", 'JSTrader', 'Ou812', format='csv'),
            parse.response=FALSE)
```

```
## [1] "USD/CAD,1354491068832,0.99,407,0.99,420,0.99297,0.99461,0.99368"
## [2] "EUR/JPY,1354491096674,107.,173,107.,181,106.912,107.189,107.019"
## [3] ""
```

`ParseTrueFX()` can be used on any of the three formats. It will determine which format it has been given, and parse it appropriately.

The `pretty` argument to `ParseTrueFX()` `ParseTrueFX()` first splits the input into a `list` of unprocessed character vectors:

```
QueryTrueFX(ConnectTrueFX('USD/CAD,EUR/JPY', 'JSTrader', 'Ou812'),
            parse.response=TRUE, pretty=FALSE)
```

```
## $Symbol
## [1] "USD/CAD" "EUR/JPY"
##
## $BidBigNumber
## [1] "0.99" "107."
##
## $BidPip
## [1] "407" "173"
##
## $OfferBigNumber
## [1] "0.99" "107."
##
## $OfferPip
## [1] "420" "181"
##
## $High
## [1] "0.99461" "107.189"
##
## $Low
## [1] "0.99297" "106.912"
##
## $TimeStamp
## [1] "1354491077498" "1354491096674"
```

Then, if called with `pretty=TRUE` (the default), it will convert the `list` to a `data.frame` – like those in the previous examples – converting strings to `numeric` or `POSIXct` as necessary.

The `qualifier` argument to `ConnectTrueFX()` An authenticated session must have a `qualifier` which can be any alphanumeric string. The default value for the `qualifier` argument to `ConnectTrueFX()` is `"default"`, although you're free to use something else.

```
con <- ConnectTrueFX('USD/JPY,EUR/JPY', 'JSTrader', 'Ou812',
                     qualifier="2yen")
con$qualifier
```

```
## [1] "2yen"
```

TFXsession objects

A `TFXsession` object is just an environment that contains the info needed by `QueryTrueFX()`. There is a `print.TFXsession` method in version 0.1.1 of **TFX**. If you are using version 0.1.0, your output will look different. The `print.TFXsession` method will only print "non-private" info. It will not print username or password.

```
eurss
```

```
## <TFXsession 1354491095901>
##  $ currency.pairs:"EUR/USD"
##  $ qualifier     :"default"
##  $ active        :FALSE
##  $ snapshot      :TRUE
##  $ format        :"default"
##  $ last.used     :"2012-12-02 17:31:35.884 CST"
```

You can explore a `TFXsession` object just like any other `environment`: `ls`, `ls.str(eurss)`, `as.list(eurss)`, etc.

```
ls(eurss)
```

```
##  [1] "active"         "currency.pairs" "format"         "id"
##  [5] "last.used"      "password"       "qualifier"      "session"
##  [9] "snapshot"       "URL"            "username"
```

TFX and Shiny

RStudio's shiny is *awesome*. If you haven't yet played with it yet, you should check it out (http://www.rstudio.com/shiny/).

You can install shiny by entering these commands:

```
options(repos = c(RStudio = "http://rstudio.org/_packages", getOption("repos")))
install.packages("shiny")
```

I put together a really simple shiny app that will open a browser and display real-time quotes that update every 750 milliseconds.

You can run the app with:

```
library(shiny)
runGist("4122626")
```

This shiny app uses an unauthenticated session which means that it always displays the same 10 currency pairs. However, you can easily modify the code to use an authenticated session using your login credentials. It could also be modified to update more often and/or show milliseconds in the timestamps. The code for this shiny app can be viewed or downloaded from http://gist.github.com/4122626

Historical Data

In addition to real-time quotes, TrueFX(tm) also provides historical tick data going back to 2009 for 15 currency pairs.

From the TrueFX(tm) website:

> TrueFX is the first service that brings you real, dealable prices from real market participants from all the major market makers, with absolutely no intermediary. As a technology company, we can offer you historical tick-by-tick market data, at zero cost to you.
>
> This data is top-of-the-book, tick-by-tick market data, with fractional pip spreads in millisecond detail. All timestamps are based on GMT.

The data can be downloaded from http://www.truefx.com/?page=downloads

---

*The **TFX** package is in no way affiliated, endorsed, or approved by TrueFX(tm), Integral Development Corp, or any of its affiliates. TrueFX(tm) is a brand name that is owned by Integral Development Corp.*