IMPORTING AND MANAGING FINANCIAL DATA IN R

Setting default arguments for `getSymbols()`

`getSymbols()` "methods"

- `getSymbols()` doesn't contain code to import data
- Code for each data source are in a `getSymbols.[source]` "method"
- For example:

```
> # You call getSymbols()
> getSymbols("GDP", src = "FRED")

> # getSymbols() calls source "method"
> getSymbols.FRED("GDP")
```

- Users should not call `getSymbols` "methods" directly

Use `setDefaults()` to change default data source

```
> setDefaults(getSymbols, src = "FRED")

> # Did not set src = "FRED"
> gdp <- getSymbols("GDP", auto.assign = FALSE)

> # Note the 'src' attribute
> str(gdp)
 An 'xts' object on 1947-01-01/2016-10-01 containing:
   Data: num [1:280, 1] 243 246 250 260 266 ...
  - attr(*, "dimnames")=List of 2
   ..$ : NULL
   ..$ : chr "GDP"
   Indexed by objects of class: [Date] TZ: UTC
   xts Attributes:
 List of 2
  $ src    : chr "FRED"
  $ updated: POSIXct[1:1], format: "2017-02-13 08:46:50"
```

`setDefaults()`

- Sets new default arguments using `name = value` pairs
- Only alters behavior for `getSymbols()`
- Stores values in global `options()`

Other arguments

- Find formal arguments for a `getSymbols()` source method
  - Use `args(): args(getSymbols.yahoo)`
  - Use `help(): help("getSymbols.yahoo")`

Default `from` and `to` values

```
> args(getSymbols.yahoo)
 function (Symbols, env, return.class = "xts", index.class = "Date",
     from = "2007-01-01", to = Sys.Date(), ...)
```

```
> setDefaults(getSymbols.yahoo, from = "2016-01-01", to = "2016-12-31")

> aapl <- getSymbols("AAPL", auto.assign = FALSE)

> str(aapl)
 An 'xts' object on 2016-01-04/2016-12-30 containing:
   Data: num [1:252, 1:6] 102.6 105.8 100.6 98.7 98.6 ...
  - attr(*, "dimnames")=List of 2
   ..$ : NULL
   ..$ : chr [1:6] "AAPL.Open" "AAPL.High" "AAPL.Low" "AAPL.Close" ...
   Indexed by objects of class: [Date] TZ: UTC
   xts Attributes:
 List of 2
  $ src    : chr "yahoo"
  $ updated: POSIXct[1:1], format: "2017-02-13 08:46:50"
```

getDefaults()

```
> getDefaults()
 [1] "getSymbols.yahoo"
```

```
> getDefaults(getSymbols.yahoo)
 $from
 [1] "'2016-01-01'"

 $to
 [1] "'2016-12-31'"
```

- Values returned *do not* imply those functions accept user-specified defaults

```
> setDefaults(load, file = "my_file.RData")
> getDefaults(load)  # Will not alter behavior
 $file
 [1] "'my_file.RData'"
```

IMPORTING AND MANAGING FINANCIAL DATA IN R

Let’s practice!

IMPORTING AND MANAGING FINANCIAL DATA IN R

Setting per-instrument default arguments

Use `setSymbolLookup()` to set data source

```
> # Change source only
> setSymbolLookup(AAPL = "google")
```

```
> aapl <- getSymbols("AAPL", auto.assign = FALSE)

> str(aapl)  # note the 'src' attribute
An ‘xts’ object on 2007-01-03/2017-02-22 containing:
  Data: num [1:2552, 1:5] 12.3 12 12.2 12.3 12.3 ...
 - attr(*, "dimnames")=List of 2
  ..$ : NULL
  ..$ : chr [1:5] "AAPL.Open" "AAPL.High" "AAPL.Low"
"AAPL.Close" ...
  Indexed by objects of class: [Date] TZ: UTC
  xts Attributes:
List of 2
 $ src    : chr "google"
 $ updated: POSIXct[1:1], format: "2017-02-23 14:16:55"
```

Use `setSymbolLookup()` to set other arguments

```
> setSymbolLookup(MSFT = list(src = "google", from = "2016-01-01"))
```

```
> msft <- getSymbols("MSFT", auto.assign = FALSE)

> str(msft)  # note the 'src' attribute and first date
An ‘xts’ object on 2016-01-04/2017-02-27 containing:
  Data: num [1:290, 1:5] 54.3 54.9 54.3 52.7 52.4 ...
 - attr(*, "dimnames")=List of 2
  ..$ : NULL
  ..$ : chr [1:5] "MSFT.Open" "MSFT.High" "MSFT.Low"
"MSFT.Close" ...
  Indexed by objects of class: [Date] TZ: UTC
  xts Attributes:
List of 2
 $ src    : chr "google"
 $ updated: POSIXct[1:1], format: "2017-02-23 14:20:21"
```

Save and restore defaults (1)

```
> # Set default
> setSymbolLookup(AAPL = "google")
```

```
> # Verify the default changed
> getSymbolLookup()
$AAPL
$AAPL$src
[1] "google"
```

```
> # Save lookup
> saveSymbolLookup("symbol_lookup.rda")
```

```
> # Remove lookup
> setSymbolLookup(AAPL = NULL)
```

Save and restore defaults (2)

```
> # Verify the default is removed
> getSymbolLookup()
named list()
```

```
> # Load lookup
> loadSymbolLookup("symbol_lookup.rda")
```

```
> # Verify the default is restored
> getSymbolLookup()
$AAPL
$AAPL$src
[1] "google"
```

IMPORTING AND MANAGING FINANCIAL DATA IN R

Let’s practice!

IMPORTING AND MANAGING FINANCIAL DATA IN R

Handling instrument symbols that clash or are not valid R names

Syntactically valid names

- Valid names contain letters, numbers, ., and _
- Must start with a letter, or a . followed by a non-number
- May not be one of the reserved words
- Not valid:
  - `.4times`, `_one`, `for`

Accessing objects with non-syntactic names (1)

- `getSymbols()` makes some names valid
  - S&P 500 Index: `"^GSPC"`

```
> getSymbols("^GSPC")
[1] "GSPC"

> head(GSPC, 3)
           GSPC.Open GSPC.High GSPC.Low
2007-01-03   1418.03   1429.42  1407.86
2007-01-04   1416.60   1421.84  1408.43
2007-01-05   1418.34   1418.34  1405.75
           GSPC.Close GSPC.Volume GSPC.Adjusted
2007-01-03    1416.60  3429160000       1416.60
2007-01-04    1418.34  3004460000       1418.34
2007-01-05    1409.71  2919400000       1409.71
```

Accessing objects with non-syntactic names (2)

- Some ticker symbols are not valid names
  - Shanghai Stock Exchange Composite Index: `"000001.SS"`

```
> getSymbols("000001.SS", auto.assign = TRUE)
[1] "000001.SS"

> str(000001.SS)
Error: unexpected symbol in "str(000001.SS"
```

Accessing objects with non-syntactic names (3)

```
> head(`000001.SS`, n = 3)
            000001.SS.Open   000001.SS.High     000001.SS.Low
2007-01-04         2715.72          2715.72           2715.72
2007-01-05         2641.33          2641.33           2641.33
2007-01-08         2707.20          2707.20           2707.20
           000001.SS.Close 000001.SS.Volume 000001.SS.Adjusted
2007-01-04         2715.72                0            2715.72
2007-01-05         2641.33                0            2641.33
2007-01-08         2707.20                0            2707.20
```

```
> head(get("000001.SS"), n = 3)
            000001.SS.Open   000001.SS.High     000001.SS.Low
2007-01-04         2715.72          2715.72           2715.72
2007-01-05         2641.33          2641.33           2641.33
2007-01-08         2707.20          2707.20           2707.20
           000001.SS.Close 000001.SS.Volume 000001.SS.Adjusted
2007-01-04         2715.72                0            2715.72
2007-01-05         2641.33                0            2641.33
2007-01-08         2707.20                0            2707.20
```

Valid name for one instrument

- Assign `getSymbols()` output to valid name
- Convert column names to valid names

```
> sse <- getSymbols("000001.SS", auto.assign = FALSE)
```

```
> colnames(sse) <- paste("SSE",
                         c("Open", "High", "Low", "Close",
                           "Volume", "Adjusted"), sep = ".")
> head(sse, n = 3)
            SSE.Open    SSE.High     SSE.Low
2007-01-04   2715.72     2715.72     2715.72
2007-01-05   2641.33     2641.33     2641.33
2007-01-08   2707.20     2707.20     2707.20
           SSE.Close  SSE.Volume  SSE.Adjusted
2007-01-04   2715.72           0       2715.72
2007-01-05   2641.33           0       2641.33
2007-01-08   2707.20           0       2707.20
```

Valid name for multiple instruments

- Create symbol-to-R-object mapping with `setSymbolLookup()`

```
> setSymbolLookup(SSE = list(name = "000001.SS"),
                  FORD = list(name = "F"))
> getSymbols(c("SSE", "FORD"))
[1] "SSE"  "FORD"
```

```
> head(SSE, n = 3)
             SSE.Open  SSE.High  SSE.Low  SSE.Close  SSE.Volume  SSE.Adjusted
2007-01-04    2715.72   2715.72  2715.72    2715.72           0       2715.72
2007-01-05    2641.33   2641.33  2641.33    2641.33           0       2641.33
2007-01-08    2707.20   2707.20  2707.20    2707.20           0       2707.20

> head(FORD, n = 3)
           FORD.Open  FORD.High FORD.Low  FORD.Close  FORD.Volume  FORD.Adjusted
2007-01-03      7.56       7.67      7.44        7.51     78652200       6.150263
2007-01-04      7.56       7.72      7.43        7.70     63454900       6.305862
2007-01-05      7.72       7.75      7.57        7.62     40562100       6.240346
```

IMPORTING AND MANAGING FINANCIAL DATA IN R

Let’s practice!