Package: alphavantagepf (via r-universe)

November 4, 2026

Title 'Alphavantage Financial Data' API R Wrapper and Shiny Interface

Version 0.9.22

Description Download, manage, and visualize via Shiny App 'Alphavantage financial data' <https://www.alphavantage.co/documentation/>. Data is downloaded into `data.table`s using one parameterized function. Results can be piped to optional helper functions to extract and simplify more complex data. A Shiny interface is also provided to download, manage, analyze and visualize market data.

License GPL (>= 3)

Encoding UTF-8

Depends R (>= 4.1.0)

Imports dplyr (>= 1.0.0), tidyr (>= 0.6.3), data.table, httr (>= 1.2.1), jsonlite (>= 1.5), purrr (>= 1.0), readr (>= 1.1.1), stringr (>= 1.2.0), qlcal (>= 0.1.0), lubridate (>= 1.0.0), utils (>= 4.0.0), fst (>= 0.9.8), arrow (>= 25.0.0), progress (>= 1.2.0), rlang (>= 1.2.0), TTR (>= 0.24.0), splines (>= 4.6.0), patchwork (>= 1.3.0), progressr (>= 1.0.0), fs (>= 2.0.0), usethis, ggplot2 (>= 3.0.0), FinanceGraphs (>= 0.9.0), shiny (>= 1.0.0), shinyjs (>= 2.1.0), shinyFeedback (>= 0.4.0), dygraphs (>= 1.1.0), gt (>= 1.3.0), gtExtras (>= 0.6.0), clipr (>= 0.8.0), shinychat (>= 0.5.0), mcptools (>= 1.0.0), bslib (>= 0.12.0), ellmer (>= 0.5.0), promises (>= 1.5.0)

Suggests testthat, knitr, rmarkdown

Roxygen list(markdown = TRUE)

LazyData true

VignetteBuilder knitr

URL https://github.com/derekholmes0/alphavantagepf, https://derekholmes0.github.io/alphavantagepf/

BugReports https://github.com/derekholmes0/alphavantagepf/issues

Config/roxygen2/version 8.1.1

Config/pak/sysreqs
libcairo2-dev cmake libfontconfig1-dev libfreetype6-dev git make libgit2-dev libicu-dev libpng-dev libuv1-dev libxml2-dev libssl-dev python3 libnode-dev libx11-dev zlib1g-dev

Repository https://derekholmes0.r-universe.dev

Date/Publication 2026-11-03 21:21:27 UTC

RemoteUrl https://github.com/derekholmes0/alphavantagepf

RemoteRef HEAD

RemoteSha a851c00e486bdcce8b4817a3eb106f4eeebae4fa

Contents

alphavantagepf . . . 3
av_add_analytic . . . 3
av_add_assetgroups . . . 5
av_add_earn . . . 6
av_add_options . . . 7
av_add_px . . . 8
av_extract_df . . . 10
av_funhelp . . . 11
av_get_pf . . . 12
av_grep_opts . . . 14
av_load_shinydata . . . 16
av_make_funcmap . . . 17
av_opt_helper_cols . . . 17
av_reset_defaults . . . 19
av_runShiny . . . 20
av_runShiny_addFunctions . . . 22
av_state_interface . . . 22
avpf_api_key . . . 24
avpf_set_request_pace . . . 25
avsh_clipboard . . . 25
avsh_quick_message . . . 26
avsh_set_tabtitle . . . 26
gt.basetheme . . . 27
message_if . . . 28

Index **29**

`alphavantagepf` *alphavantagepf: A lightweight R interface to the Alpha Vantage API*

Description

An R interface to the Alpha Vantage API which emphasizes normalized data nd data.table conventions

Details

The `alphavantagepf` package provides a lightweight interface to the Alpha Vantage API. Alpha Vantage is a free source for financial data that in many cases is more accurate than Yahoo Finance and Google Finance. Get a free API KEY at https://www.alphavantage.co. Then use the R interface functions `avpf_api_key("YOUR_KEY")` to set the API key and the `av_get_pf()` function to get financial data.

`av_add_analytic` *av_add_analytic*

Description

Adds a user-defined function to the av Shiny app

Usage

```
av_add_analytic(
  runcode,
  func_name,
  helpstr = "user function",
  focus = "MAIN",
  data_req = "pxd;earn;earnest",
  message_str = "",
  delay_save_state = FALSE
)
```

Arguments

`runcode`	Code string user must run to call the function.
`func_name`	Name of function run when analytic is called. **If an empty string is supplied, the runcode will be de-registered.**
`helpstr`	(default: "user function"): A string comment to ad to the av.h (help) command
`focus`	(default: "MAIN") String with tab name to set focus to when command is run
`data_req`	(default: "pxd;earn;earnest") A semicolon list of required data tables. Used for Lazy data loading.

`message_str` (default "") Additional string to add to output messages

`delay_save_state` (default: FALSE) Do not save state to cache files. Used for speed optimization

Details

Adds or removes a new command to the av_runShiny app

When the `av_runShiny()` app is run, users can call functions to provide analytics based on asset strings in the command line. This function allows users to add their own analytics by registering a function which takes, as inputs

1. `todo`: The command line and any subsequent parameters as a space delimited string
2. `rv`: Reactive values supplied by the Shiny app. In particular the parameter `rv$istr1` contains the semicolon delimited set of assets prior to the command invocation. The registered function should return a (possibly named, see vignette) list containing one or more `gt()` tables, `dygraphs()`, or `ggplots()` to be displayed when the command is run. See vignette for specfic details

- The function specified must be available (i.e in `.GlobalENv()`) to the Shiny app when the command is run. Otherwise an error message will be displayed.
- If the specified command has already been registered, a message will be given and the internal data will be overridden.

Value

String message with success or failure of function addition.

See Also

`av_runShiny()`

Examples

```
## Not run:
my_testfunc <- function(todo,rv) {
  message("todO: ",todo," with asset string ",rv$istr1)
  n_to_return <- c(strsplit(todo," "),"3")[[2]] |> as.numeric()
  table1 <- head(mtcars,n_to_return) |> gt()
  table2 <- data.table(asset=strsplit(rv$istr1,";")) |> gt()
  plot1 <- ggplot(mtcars,aes(mpg,disp)) + geom_point()
  return(list(table1, table2, plot1))
}
av_add_analytic("TEST","my_testfunc",helpstr="a test func")
# From the app; run "QQQ;SPY test 5"
# From the app: run "av.h"

## End(Not run)
```

av_add_assetgroups *Add asset lists*

Description

Adds asset lists to av_runShiny() internal data.

Usage

```
av_add_assetgroups(indta)
```

Arguments

indta	A data.frame with a minimum of two columns c("listnm","ticker") with one or more lines for each "listnm" and possibly a column weight for weightings

Details

Add Asset Groups

Lists are specified in normalized form. Duplicate list names with those currently in use are replaced.

Value

Nothing

See Also

av_runShiny()

Examples

```
## Not run:
newtickers <- c("QQQ","QQQE","NDX")
av_add_assetgroups(data.table(listnm=rep("nasdaq",length(newtickers)),ticker=newtickers))
# To remove an asset list, just use an empty string for the ticker
av_add_assetgroups(data.table(listnm=c("new"),ticker=c("")))

## End(Not run)
```

av_add_earn *Add or download Earnings Data*

Description

Adds earnings data to av_runShiny() internal data, either by download or with user data

Usage

```
av_add_earn(
  substitute_earn = NULL,
  substitute_earnest = NULL,
  equitylist = NULL,
  delay = 0
)
```

Arguments

substitute_earn	A (default NULL) data.frame with past earnings
substitute_earnest	(default NULL) A data.frame with earnings estimates
equitylist	(default NULL) A list with tickers for which to retrieve earnings (from AlphaVantage)
delay	(default 0) A numeric value specifying delay between Alphavantage calls (in seconds)

Details

Add Earnings Data

Entire set of columns from av_get_pf() can be added. First date column renamed to timestamp. If just assetypes is given, the function downloads earnings as needed (respecting maximum age parameters defined in the app's AVOPTS tab.) **Note that price data must always be added first**

Value

Data.table with summary of downloaded or added earnings

See Also

av_runShiny()

Examples

```
## Not run:
# To add earnings for a set of tickers
av_load_shinydata()  # Make sure most recent data is loaded
av_add_earn(equitylist=data.table(symbol=c("IBM","GS")))

## End(Not run)
```

---

av_add_options *av_add_options*

Description

Manages option database and creates vol surfaces

Usage

```
av_add_options(
  todo,
  dtstr = "-1w::",
  symbols = NULL,
  freq = "d",
  replace_data = FALSE,
  baddates_limits = 3,
  verbosity = "basic",
  external_path = NULL
)
```

Arguments

todo	What to do to add options. Choices are in the table below

todo	Description
update	Download and add or update full option runs by symbol over periods in dtstr
inventory	Create/update inventory of options that have been downloaded
reconstruct_inventory	Reconstruct inventory of all options that have been downloaded
iv	Updates internal data.table (eqopt_iv of Implied term structures by symbol and date
copy_external	Copies identical directory structures from external_path to siny app internal data

dtstr	(default "-1w::") How far back to load options. ONly data needed within time-frame is downloaded, unless replace_data=TRUE
symbols	(default: NULL) List of symbols to download
freq	(Default "d") Granularity of data downloaded ("d" for daily, "w" for weekly, "m" for monthly)
replace_data	(Default FALSE) Update or replace data

`baddates_limits`
: (default 3): Consecutive business days of null data beyond which options are assumed not to exist.

`verbosity`
: (default `"basic"`) What to display as data is added. Options may be comma delimited and are

verbosity	Description
basic	Most basic information
timing	Timing information
iv	Implied volatility surface calculation progress
ivprobs	Implied volatility surface calculation errors (e.g. not enough data, etc.)

`external_path`
: (default NULL) Path from which to replace existing data

Details

Manage option data

This function can take a long time to run. Full option quotes are stored by symbol in arrow/parquet format by symbol day, and contract.

- Unless `replace_data==TRUE`, only dates and symbols that need to be downloaded will.
- Except the `OS S` function, all analytics run in the app require options data to be downloaded using this function.

Value

nothing

See Also

`av_runShiny()`

---

av_add_px *Add or download Price or Time Series Data*

Description

Adds price data to `av_runShiny()` internal data.

Usage

```
av_add_px(indta = NULL, assettypes = NULL, equitylist = NULL, dtstr = "-30y::")
```

Arguments

indta	(default: NULL) A data.frame with the following minimal columns: c(symbol,timestamp,close). Other variables added could be c(adjusted_close,open,high,low,volume,dividend_amount,split If adjusted_close is not in the dataset, it will be set to close
assettypes	(default NULL) An optional data.frame with minimal columns c(symbol,type,currency,name) with descriptive data for the assets given in indta. If not specified, a call to av_get_pf(.,"SYMBOL_SEARCH") is necessary to determine the asset type (one of c("Equity","ETF","FX","Index","Crypto")) for subsequent calls to av_get_pf()
equitylist	(default NULL) If specified, function will get equity prices from av_get_pf. indta can be null or is otherwise ignored.
dtstr	(default "-30y::"). Date range to download if applicable.

Details

Entire set of columns from av_get_pf() can be added. First date column renamed to timestamp internally.

Value

Nothing

See Also

av_runShiny()

Examples

```
## Not run:
# To add known symbols outside the app
av_load_shinydata()  # Make sure most recent data is loaded
av_add_px(equitylist=c("IBM","GS","JPM"))

# To add ad-hoc data from Alphavantage (e.g. Natgas spot at Henry Hub)
# Note that "symbol" in indta must match same in assettypes
asset_df <- data.frame(symbol=c("GAS_HH"),type=c("user"),currency=c("USD"), name=c("GAS_HH"))
ng_data <- av_get_pf("","NATURAL_GAS")[,.(symbol="GAS_HH",timestamp,close=value)]
av_add_px(ng_data, assettypes=asset_df)

# To data from other sources
suppressMessages(require(quantmod))
ffdta <- as.data.table(quantmod::getSymbols("FEDFUNDS",src="FRED",auto.assign=FALSE))
ffdta <- ffdta[,.(DT_ENTRY=index,close=FEDFUNDS,adjusted_close=FEDFUNDS,symbol="FEDFUNDS")]
av_add_px(ffdta)

## End(Not run)
```

av_extract_df *Extract data from AlphaVantage API results*

Description

av_extract_df() pulls out nested data.frames from mixed data returned by av_get_pf() av_extract_fx() returns a simplified FX quote in data.table formfrom av_get_pf() calls. av_extract_analytics() returns melted data.table from calls to av_get_pf("ANALYTICS_FIXED_WINDOW") or av_get_pf("ANALYTICS_SLIDING_WIN av_extract_divs_or_splits() returns melted data.table from calls to av_get_pf("DIVIDENDS") or av_get_pf("SPLITS")

Usage

```
av_extract_df(indta, grepstring = "", melt = FALSE, empty_dt_onerror = TRUE)

av_extract_fx(indta, outputform = "common", cols = "", empty_dt_onerror = TRUE)

av_extract_analytics(indta, separate_vars = FALSE, empty_dt_onerror = TRUE)

av_extract_divs_or_splits(indta, empty_dt_onerror = TRUE)
```

Arguments

indta	A data.table as returned by av_get()
grepstring	select which variable (data item) to unnest in data.table returned from av_get_pf
melt	Return data in melted/normalized form
empty_dt_onerror	(default : TRUE): Return gracefully an empty data.table if requested item is not present.
outputform	(default : common, av_extract_fx() only): Use common names from REALTIME_BULK_QUOTES
cols	(default : all columns: av_extract_fx() only): String or List of columns to return
separate_vars	(default : FALSE) separate out multiple levels of variable names into new keys

Details

av_get_pf() frequently returns a nested data.table, or a structure with nested data.frames. These are utilities functions to extract, filter and summarize returned values. if av_get_pf() returns a valid response, but empty extracted data.tables, am empty data.table() will be returned NOTE: (Change 20260724: Timestamp is returned in POSIXct form in local Sys.timezone() time)

Value

Extracted data.tables for nested data returned from av_get_pf(), If grepstring is not specified, first nested table is returned. av_extract_fx() returns a shortened data.table with FX quotes.

See Also

av_get_pf(), av_grep_opts()

Examples

```
## Not run:
av_get_pf("","MARKET_STATUS")  |> av_extract_df()
av_get_pf("","TOP_GAINERS_LOSERS") |> av_extract_df("top_losers")
av_get_pf("USD/BRL","CURRENCY_EXCHANGE_RATE") |> av_extract_fx()
av_get_pf(c("ORCL","IBM"),"ANALYTICS_FIXED_WINDOW") |> av_extract_analytics(separate_vars=TRUE)

## End(Not run)
```

---

av_funhelp *Extract data from Alpha Vantage retuned data*

Description

Extract data from Alpha Vantage retuned data

Usage

```
av_funhelp(av_fun_grep = "", verbose = TRUE)
```

Arguments

av_fun_grep	A Alpha Vantage function name or portions of one. For a list of parameters, visit the Alpha Vantage API documentation.
verbose	(Default: TRUE) Prints the help string

Details

Returns defaults and parameter lists for Alphavantage functions

Value

Help Text and default parameters.

See Also

av_get_pf()

Examples

```
av_funhelp("GLOBAL_QUOTE")
```

av_get_pf *Get financial data from the Alpha Vantage API*

Description

Interface to alphavantage API.

Usage

```
av_get_pf(
  symbol,
  av_fun,
  symbolvarnm = "symbol",
  dfonerror = TRUE,
  melted = "default",
  delay = 0,
  verbose = FALSE,
  ...
)
```

Arguments

symbol	A character string of an appropriate stock, fund, or currency See parameter "symbol" in Alpha Vantage API documentation.
av_fun	A character string matching an appropriate Alpha Vantage "function". See parameter "function" in Alpha Vantage API documentation.
symbolvarnm	(default: symbol) Variable name which has the symbol requested. Set to a blank string if not wanted.
dfonerror	(default: TRUE) Return an empty data.table when any error occurs
melted	(default: "default") String specifying when to melt, "default" is chosen by the package, "TRUE|always" always melt, "FALSE|never" never melts
delay	(default: 0) Delay in seconds after API call, used to embed within large single-symbol calls. **If delay <=0** then delays are scheduled to keep within the request pace specified in avpf_set_request_pace().
verbose	(default: FALSE) Print debug information helpful for errors. Also copies full url to clipboard.
...	Additional parameters or overrides passed to the Alpha Vantage API. For a list of parameters, visit the Alpha Vantage API documentation.

Details

The av_fun **argument replaces the API parameter "function"** because function is a reserved name in R. All other arguments match the Alpha Vantage API parameters.

There is no need to specify the `apikey`**,** `datatype`**,** `outputsize` **or** `entitlement` **parameters** as arguments to av_get_pf(). Before using, you must set the API key using av_api_key("YOUR_API_KEY"). `outputsize` defaults to "full" unless overridden with "compact in ...."

Required parameters other than `symbol` must be passed as named arguments via ....

Optional parameters have defaults which can be obtained by calling `av_funhelp()` and overridden via ....

There is no need to specify the datatype parameter as an argument to `av_get_pf()`. The function will return a data.table.

ForEx "FROM/TO" symbol details. FOREX symbols in the `av_get_pf()` function are supplied in `"FROM/TO"` format, which are then parsed in the Alpha Vantage API into `from_currency` and `to_currency` API parameters. Usage example: `av_get_pf("USD/BRL", "FX_DAILY")`

Value

Returns a data.table with results dependent on the function called. Mixed data is returned as a melted data.table, possibly with nested data.frames. Time series are returned as data.tables.

See Also

`avpf_api_key()`, `av_extract_df()`, `av_extract_fx()`, `av_grep_opts()`,`av_funhelp()`, `avpf_set_request_pace()`

Examples

```
## Not run:
avpf_api_key("YOUR_API_KEY","delayed") # or "realtime" if you have such access

# example code

# ---- 1.0 SINGLE NAME EQUITY SUMMARY INFORMATION AND SEARCH ----

av_get_pf("IBM","OVERVIEW") |> str()
av_get_pf("EWZ","ETF_PROFILE")
av_get_pf("EWZ","ETF_PROFILE") |> av_extract_df("holdings")
av_get_pf("","SYMBOL_SEARCH",keywords="COMMERCE")

# ---- 2.0 MARKET QUOTES  ----

av_get_pf("IBM","GLOBAL_QUOTE")
av_get_pf("USD/BRL","CURRENCY_EXCHANGE_RATE") |> av_extract_fx()
av_get_pf(c("ORCL","IBM","EWZ","ARGT"),"REALTIME_BULK_QUOTES",melt=FALSE)
# Note you need advanced permissioning for REALTIME_BULK_QUOTES

# ---- 3.0 SINGLE NAME HISTORICAL DATA  ----

av_get_pf("IBM","TIME_SERIES_DAILY")
av_get_pf("IBM","TIME_SERIES_INTRADAY")

# ---- 4.0 MARKET PRICING DATA  ----

av_get_pf("","MARKET_STATUS")  |> av_extract_df()
```

```
av_get_pf("","TOP_GAINERS_LOSERS") |> av_extract_df("top_losers")
av_get_pf("","TREASURY_YIELD",maturity='7year')

# ---- 4.0 SINGLE NAME NON-PRICING DATA  ----

av_get_pf("IBM","DIVIDENDS")
av_get_pf("IBM","EARNINGS")  |> av_extract_df("quarter",melt=TRUE)
av_get_pf("IBM","NEWS_SENTIMENT") |> av_extract_df("feed")

av_get_pf("IBM","EARNINGS_CALL_TRANSCRIPT",quarter="2024Q3") |> av_extract_df("transcript")
 # Note that quarter is a required parameter, not specifying will throw an error

 # ---- 5.0 SINGLE NAME OPTION PRICING DATA  ----

av_get_pf("IBM","HISTORICAL_OPTIONS") |> av_grep_opts("F,M,put",mindays=2)

# ---- 6.0 TECHNICAL INDICATORS  ----

av_funhelp("SMA")  # Shows parameters and defaults chosen by this package.
av_get_pf("IBM","SMA",time_period=20)

# ---- 7.0 WINDOW ANALYTICS  ----

av_get_pf(c("ORCL","IBM","EWZ","ARGT"),"ANALYTICS_FIXED_WINDOW",verbose=TRUE) |>
            av_extract_analytics(separate_vars=TRUE)

## End(Not run)
```

---

av_grep_opts *Filter options from returned AV data*

Description

Filter options from returned AV data

Usage

```
av_grep_opts(
  indta,
  grepstring = "F,M,C,otm",
  spot = NULL,
  mindays = 3,
  startdt = Sys.Date(),
  mindelta = 0.05,
  dropsymbol = FALSE
)
```

Arguments

indta	A data.table as returned by av_get_pf()
grepstring	(default F,M,call,otm,act) Three to five item string to select specific maturities, option types, strike ranges, and open interest categories from option sets returned by av_get_pf(.,"HISTORICAL_OPTIONS"). Each item in the list is an abbreviated code for what to select. The items are not case sensitive and in order are

Item	Values	description
Expiration Limit	F|B|A	F is first maturity, B is second maturity, A (pr blank) are all
Maturity Type	M|Q|A	M for monthly contracts, Q for quarterly, A (pr blank) are all
Option Type	C|P|A	Calls, puts, all (or blank) for all
Moneyness	otm|itm|A	Out of the money, in the money or both
Activity	act|A	If A, only pass contracts with positive open interest

spot	(default NULL) Spot to be used to determine itm/otm, If null then it is inferred from most out of the money call or put Note: This parameter only applies if there is one symbol in indta. If there is more than one ticker in indta a column spot must be in indta to get correct results.
mindays	(default 3). Minimum number of days to expiration to be passed through from startdt
startdt	(default Sys.Date()). Date from which expirations will be considered.
mindelta	(default 0.05) delta limit on both moneyness sides, i.e. pass only options with deltas in range c(mindelta,1-mindelta)
dropsymbol	(default FALSE). Drop symbol from returned data table.

Details

Alphavantage doesn't always return quarterly options

av_get_pf() returns a large list of options. This function helps to narrow down the list by maturity and moneyness.

Value

A reduced set of options obtained from av_get_pf() using Alphavantage HISTORICAL_OPTIONS function.

See Also

av_get_pf()

Examples

```
## Not run:
av_get_pf("IBM","HISTORICAL_OPTIONS") |> av_grep_opts("F,M,put",mindays=2)
```

```
## End(Not run)
```

`av_load_shinydata` *Load av_runShiny() internal data.*

Description

Loads internal data (prices, earnings, etc.

Usage

```
av_load_shinydata(item = NULL, verbose = TRUE, addfuncs = TRUE)
```

Arguments

`item`	Any data name as seen by running `dump_state()`. **If blank, loads entire database**
`verbose`	(default TRUE) write a status message to console
`addfuncs`	(default TRUE) Readd and re-source any user defined functions
`restore_lazy`	(default FALSE) Also restore all lazy data

Details

Load ShinyApp data from Cache

Value

Data item specified by `item` or a nothing (but a message) if left blank

See Also

`av_runShiny()`

av_make_funcmap *Extract data from Alpha Vantage retuned data*

Description

Extract data from Alpha Vantage retuned data

Usage

```
av_make_funcmap()
```

Details

Creates a data.table with API signatures and default values for Alphavantage API calls

Value

function map dat.table, also written to data directory

See Also

av_get_pf()

Examples

```
## Not run:
av_make_funcmap()

## End(Not run)
```

av_opt_helper_cols *Add additional columns to returned option sets*

Description

Add additional columns to returned option sets

Usage

```
av_opt_helper_cols(indta, scaling = NULL, spot_override = NULL)
```

Arguments

indta	An option data.table as returned by av_get_pf()
scaling	(default NULL) Scaling factor for marks, last values, and greeks. Options are • NULL or "none" : No Scaling • "10contracts" : 10 contracts converted into market value • "10kMV" : 10,000 USD converted into equivalent market value • number : Any numeric value in thousands of USD into equivalent market value
spot_override	(default NULL) Spot to be used to determine itm/otm, If null then it is inferred from most out of the money call or put Note: This parameter only applies if there is one symbol in indta. If there is more than one ticker in indta a column spot must be in indta to get correct results.

Details

Adds columns including

Column	Definition
daysExp	Days to Expiration
bo_pct	Bid offer in percent of option mid
bid_size_poi	Bid size percent of open interest
ncak	Notional number of contracts

Value

An option data.table with extra columns helpful for further analysis

See Also

av_grep_opts()

Examples

```
## Not run:
av_get_pf("IBM","HISTORICAL_OPTIONS") |> av_grep_opts("F,M,put",mindays=2) |> av_opt_helper_cols()

## End(Not run)
```

`av_reset_defaults` *Extract data from Alpha Vantage retuned data*

Description

Extract data from Alpha Vantage retuned data

Usage

```
av_reset_defaults(fileopts = TRUE, keep_apikeys = FALSE, resetgrep = "*")
```

Arguments

`fileopts`	(default: TRUE) If TRUE, then remove all files and subdirectories
`keep_apikeys`	(default: FALSE) Keep whatever API keys are stored
`resetgrep`	(default "*") Only reset grepped default variables

Details

Resets `av_runShiny()` defaults to original (newly installed) state

Value

No return

See Also

`av_runShiny()`

Examples

```
## Not run:
av_reset_defaults()

## End(Not run)
```

av_runShiny *RShiny App*

Description

av_runShiny() runs an interactive RShiny app with a professional command line interface to download, manage, and visualize data from the Alpha Vantage data service. The app treats equities, ETFs, indices, and FX equally, so users do not need to know specific API calls to integrate data. Downloaded data is cached to reduce API calls, all data can be saved for external use, and both external data and external commands (that call user functions) can be added to the app. Sets of securities can be easily added and managed.

Usage

```
av_runShiny()
```

Details

Invocation starts Shiny application. See vignettes for full details. **On first use**, In the AVOPTS tab, fill in the following fields, and hit the "Set Opts" button.

- **AV API Key**: API key obtained from Alpha Vantage
- **Entitlement** Entitlement status (either delayed or realtime) Other options that can be set are given below.

To get data and run analyses: Enter in the top yellowed line a semicolon delimited set of securities (Equity, ETF, FX, Crypto, indices) followed by a command to run analytics on those securities. Commands without securities are always prefixed by '"AV.". The results are shown as tables, plots, or dygraphs below the command line, and possibly in an additional tab to the right of the main tab. The secondary tab will change names as appropriate, and the focus may be changed off the MAIN tab.

Commands are not case sensitive, and may refer to a counterasset in the second yellowed line. A few examples include:

Command	Description
AV.H	List all available functions
AV.INV	Inventory of all available data
AV.TICKERS	Inventory of indices and crypto pairs available from Alpha Vantage
SPACE S	Search for all tickers with SPACE in their names
IBM;QQQ;NDX;USD/MXN GPI	Produce a rebased time series graph of the securities
IBM;QQQ GV	Time series graph of rolling volatiliies
ibm;qqq rv	Relative Value /Active Return Analysis
IBM;AAPL;USD/MXN Q	Table of latest prices for each instrument
USD/BRL;USD/MXN SCATI	Scatterplot of rebased levels and returns
IBM;AAPL EA	Table of Earnings data for each instrument
IBM;AAPL CN	Table of recent news items for each asset, with links
IBM;AAPL RV	Graph of excess returns over counterasset (SPY by default)

`IBM OS F,M,P,otm` Search for (e.g.) Out of the money front month puts

If an analysis requires price, dividend or earnings data that is not in the app's internal store, then it will download it as necessary from Alpha Vantage. Other data (e.g. News, options) are downloaded freshly each time. One of the key contributions of this app is to abstract out the asset-specific functions calls that are necessary to do cross-asset analyses.

Other niceties:

- Groups of assets can be named, saved, and recalled easily. The saved name can then be used in lieu of the original list. See Basic Usage vignette
- Internal data can be stored and used (as `data.table()`s) in a directory of the users' choosing. See Options vignette to set the directory, and the Data vignette for details on how the data is structured.
- Events and other plotting options can be specified in the app. See functions vignette and e.g. FinanceGraphs
- Data within each produced graph or table can be copied to the clipboard (option in `AVOPTS` page), See Setup vignette to see where to turn this on.
- Raw downloaded data can be saved into a "dump directory" (option in `AVOPTS` page)
- A current inventory list is always kept in a separate tab, avoiding having to run `AV.INV` repeatedly.
- User price data, earnings data, and asset groups can be added using helper functions `av_add_px()`, `av_add_earn()`, `av_add_assetgroups()`. See data vignette. Adding data from the console immediately adds the data to the app.
- User analytics can be added using `av_add_analytic()`. See examples in the Extensions vignette

Value

Nothing

A `ShinyAppHandle` object.

Examples

```
## Not run:
av_runShiny()

## End(Not run)
```

av_runShiny_addFunctions *Add experimental functions from a given directory*

Description

Adds analytics in code taken from single directory.

Usage

```
av_runShiny_addFunctions(fun_dir = NULL)
```

Arguments

fun_dir	Directory containing functions with input signatures to add

Details

Add a set of analytics from a code direcory

Each file will be read in the specified directory and any function with a valid signature will be added to the list of available functions in the shiny app. The function signature is a call to the function with a single argument "signature" which returns a list of four items:

1. a short name for the function,
2. the function name,
3. a help string for the function.
4. A semicolon delimited list of data tables necessary to run the function. Used for Lazy Loading If those conditions obtain, the function will be added to `av_runShiny()`

Value

Nothing

av_state_interface *av_state_interface*

Description

retrieves internal data state of `av_runShiny()` `dump_state(typegrep="*")` `dump_inv(invgrep="*")` `dump_assetgroups()` `dump_captured(todo="byfunction")` `av_shiny_px()`

Usage

```
dump_state(stategrep = "*", trunc_length = 35)

dump_inv(invgrep = "*")

dump_assetgroups()

dump_av_funcs()

dump_captured(todo = "byfunction")
```

Arguments

stategrep	: Grep string for internal state parameters, either a class or a name
trunc_length	: (default: 35) Maximumlength of character values returned.
invgrep	: A regular expression string
todo	: One of c("byfunction","pxhist",any av function name)

Details

Return av_runShiny data and states

Value

data.table with desired data.

See Also

av_runShiny()

Examples

```
## Not run:
`dump_state()`
`dump_inv()`
`dump_av_funcs()`
`dump_assetgroups()`
`dump_captured(todo="byfunction")`

## End(Not run)
```

avpf_api_key *Set the Alpha Vantage API Key*

Description

avpf_api_key() sets Alphavantage API key and entitlement code

Usage

```
avpf_api_key(api_key, entitlement = NULL)
```

Arguments

api_key	A character string with your Alpha Vantage API Key.
entitlement	An optional character string with your Alpha Vantage entitlement status. If not "delayed" or "realtime" entitlement not added to API string.

Details

The Alpha Vantage API key must be set prior to using av_get_pf(). You can obtain an API key at the Alpha Vantage Website.

Value

Invisibly returns two item list with API key and entitlement string once set). Use print method to view.

See Also

av_get_pf()

Examples

```
## Not run:
avpf_api_key("YOUR_API_KEY",entitlement="delayed")
av_get_pf("IBM", "TIME_SERIES_INTRADAY")

## End(Not run)
```

avpf_set_request_pace *Set the Alpha Vantage max requests per minute*

Description

avpf_set_request_pace() sets Alphavantage API key and entitlement code

Usage

```
avpf_set_request_pace(max_requests_per_min = 60)
```

Arguments

max_requests_per_min	(default 60) Maximum requests per minute

Details

Change this if you have a better plan with AlphaVantage

Value

Nothing

See Also

avpf_api_key()

avsh_clipboard *avsh_clipboard*

Description

Copies a data.frame to the clipboard, with a status message if relevant

Usage

```
avsh_clipboard(x, title = "")
```

Arguments

x	A data.frame or equivalent.
title	String to add to a message printed if relevant

Details

Copy data to clipboard

Value

Nothing

`avsh_quick_message` *avsh_quick_message*

Description

Displays a message underneath an input box

Usage

```
avsh_quick_message(this_message, eval = TRUE, color = "#1f78b4")
```

Arguments

`this_message`	(default "") A text message to be used. If empty string, the current message is cleared.
`eval`	(default TRUE) OPtional parameter to suppress execution.
`color`	Optional text color

Details

Display a user message in the av_runShiny app

Value

logical value of `eval`

`avsh_set_tabtitle` *avsh_set_tabtitle*

Description

Sets the title for the Details tab

Usage

```
avsh_set_tabtitle(newtext = "DETAIL", tabnm = "detail", makefocus = TRUE)
```

Arguments

`newtext`	(default"DETAIL") What to name the tab as
`tabnm`	(default "detail") inputID of relevant tab
`makefocus`	(default: TRUE) Upon setting the tab title, select the tab.

Details

Set an av_runShiny Tab Title

Value

Nothing

`gt.basetheme` *App formatting functions*

Description

Adds default elements to `gt()` table

Usage

```
gt.basetheme(
  x,
  gtopts = "all",
  sizepct = 100,
  style = 4,
  digits = 2,
  interactive_mult = 1.14,
  seps = FALSE,
  na_format = "-",
  size = "",
  interactive = ""
)
```

Arguments

`x`	Input data or `gt` table
`gtopts`	Which elements to add
`sizepct`	(default: 70) How big to make tables
`style`	(default: 4) Style number (see `gt()`)
`digits`	(default: 2) Number formatting
`interactive_mult`	(default: 80/70) Multtiple of `sizepct` applies to interactive tables.
`seps`	(default: FALSE) Number formatting thousands separators
`na_format`	(default: "-") WHat to show for NAs
`size`	(default: "") Size of text, either `""` or `"small"`
`interactive`	(default: "") String with interative elements: contains one of `(all|filter|search)`

`message_if` *av_runShiny utilities*

Description

Prints message if condition met

Usage

```
message_if(reallydothis, ...)
```

Arguments

`reallydothis`	Condition to be met
`...`	Additional items to be passed to `message`

Index

alphavantagepf, 3
av_add_analytic, 3
av_add_analytic(), *21*
av_add_assetgroups, 5
av_add_assetgroups(), *21*
av_add_earn, 6
av_add_earn(), *21*
av_add_options, 7
av_add_px, 8
av_add_px(), *21*
av_extract_analytics *(*av_extract_df*)*, 10
av_extract_df, 10
av_extract_df(), *13*
av_extract_divs_or_splits *(*av_extract_df*)*, 10
av_extract_fx *(*av_extract_df*)*, 10
av_extract_fx(), *10*, *13*
av_funhelp, 11
av_funhelp(), *13*
av_get_pf, 12
av_get_pf(), *6*, *9–11*, *13*, *15*, *17, 18*, *24*
av_grep_opts, 14
av_grep_opts(), *11*, *13*, *18*
av_load_shinydata, 16
av_make_funcmap, 17
av_opt_helper_cols, 17
av_reset_defaults, 19
av_runShiny, 20
av_runShiny(), *4–6*, *8, 9*, *16*, *19*, *22, 23*
av_runShiny_addFunctions, 22
av_state_interface, 22
avpf_api_key, 24
avpf_api_key(), *13*, *25*
avpf_set_request_pace, 25
avpf_set_request_pace(), *12, 13*
avsh_clipboard, 25
avsh_quick_message, 26
avsh_set_tabtitle, 26

dump_assetgroups *(*av_state_interface*)*, 22
dump_av_funcs *(*av_state_interface*)*, 22
dump_captured *(*av_state_interface*)*, 22
dump_inv *(*av_state_interface*)*, 22
dump_state *(*av_state_interface*)*, 22
dump_state(), *16*

gt(), *27*
gt.basetheme, 27

message_if, 28