Package 'crypto'

January 9, 2020

Type Package

Title Cryptocurrency Market Data

Description Retrieves crypto currency current and historical information as well as information on the exchanges they are listed on. For current and historical it will retrieve the daily open, high, low and close values for all crypto currencies. This retrieves the historical market data by web scraping tables provided by 'Cryptocurrency Market Capitalizations' <https://coinmarketcap.com>.

Version 1.1.3

Date 2020-01-09

Maintainer Jesse Vent <cryptopackage@icloud.com>

URL https://github.com/JesseVent/crypto,
https://CRAN.R-project.org/package=crypto

BugReports https://github.com/JesseVent/crypto/issues

Depends R (>= 3.4.0), rvest, xml2

License MIT + file LICENSE

Encoding UTF-8

LazyData true

Imports dplyr, tibble, jsonlite, lubridate, xts, curl, utils, cli, crayon, progress, stats, tidyr, httr, rstudioapi, reshape2

Suggests R6

SystemRequirements libxml2-devel, libcurl-devel, openssl-devel, libsecret-devel, libsodium-devel

RoxygenNote 6.1.1

NeedsCompilation no

Author Jesse Vent [aut, cre]

Repository CRAN

Date/Publication 2020-01-09 21:20:02 UTC

R topics documented:

`cmc_api` *CoinMarketCap Professional API Call*

Description

CoinMarketCap Professional API Call

Usage

```
cmc_api(url)
```

Arguments

`url`	CoinMarketCap API URL

Value

api results

Examples

```
## Not run:
cmc_api('https://pro-api.coinmarketcap.com/v1/cryptocurrency/listings/latest')

## End(Not run)
```

`crypto_global_market` *Global crypto currency market data*

Description

Retrieve daily snapshot of market_cap and the volume traded for either total crypto currency market or the altcoin market only. Selecting 'total' will include bitcoin and all altcoins.

Usage

```
crypto_global_market(market = NULL)
```

Arguments

`market`	Either 'total' or 'altcoin'

Value

Daily time series of token data in a dataframe:

`timestamp`	Timestamp (POSIXct)
`market_cap`	Market Cap in USD
`volume`	Volume traded in USD

Note

The API this uses often will change, please raise an issue on GitHub if errors persist.

Examples

```
## Not run:
##
market <- 'total'
crypto_global_market <- crypto_global_market(market)

## End(Not run)
```

`crypto_history` *Get historic crypto currency market data*

Description

Scrape the crypto currency historic market tables from CoinMarketCap <https://coinmarketcap.com> and display the results in a date frame. This can be used to conduct analysis on the crypto financial markets or to attempt to predict future market movements or trends.

Usage

```
crypto_history(coin = NULL, limit = NULL, start_date = NULL,
  end_date = NULL, coin_list = NULL, sleep = NULL)
```

Arguments

`coin`	string Name, symbol or slug of crypto currency, default is all tokens
`limit`	integer Return the top n records, default is all tokens
`start_date`	string Start date to retrieve data from, format 'yyyymmdd'
`end_date`	string End date to retrieve data from, format 'yyyymmdd'
`coin_list`	string Valid values are 'api', 'static' or NULL
`sleep`	integer Seconds to sleep for between API requests

Value

Crypto currency historic OHLC market data in a dataframe:

`slug`	Coin url slug
`symbol`	Coin symbol
`name`	Coin name
`date`	Market date
`ranknow`	Current Rank
`open`	Market open
`high`	Market high
`low`	Market low
`close`	Market close
`volume`	Volume 24 hours
`market`	USD Market cap
`close_ratio`	Close rate, min-maxed with the high and low values that day
`spread`	Volatility premium, high minus low for that day

This is the main function of the crypto package. If you want to retrieve ALL coins then do not pass a argument to crypto_history(), or pass the coin name.

Examples

```
## Not run:

# Retrieving market history for ALL crypto currencies
all_coins <- crypto_history(limit = 1)

# Retrieving this years market history for ALL crypto currencies
all_coins <- crypto_history(start_date = '20180101')

## End(Not run)
```

`crypto_list` *Retrieves name, symbol, slug and rank for all tokens*

Description

List all of the crypto currencies that have existed on CoinMarketCap and use this to populate the URL base for scraping historical market data. It retrieves name, slug, symbol and rank of crypto currencies from CoinMarketCap and creates URLS for `scraper()` to use.

Usage

```
crypto_list(coin = NULL, start_date = NULL, end_date = NULL,
  coin_list = NULL)
```

Arguments

`coin`	Name, symbol or slug of crypto currency
`start_date`	Start date to retrieve data from, format yyyymmdd
`end_date`	Start date to retrieve data from, format yyyymmdd
`coin_list`	'api', 'static' or NULL

Value

Crypto currency historic OHLC market data in a dataframe:

`symbol`	Coin symbol (not-unique)
`name`	Coin name
`slug`	Coin URL slug (unique)
`rank`	Current rank by market cap
`exchange_url`	Exchange market tables urls for scraping
`history_url`	Historical market tables urls for scraping

Required dependency that is used in function call `getCoins()`.

Examples

```
## Not run:
coin <- 'kin'
coins <- crypto_list(coin)

# return all coins
coin_list <- crypto_list()

## End(Not run)
```

`crypto_prices` *Get current crypto currency prices*

Description

This will retrieve the current market prices from CoinMarketCap. Data gets refreshed every 5 minutes but will only return 100 at a time.

Usage

```
crypto_prices(coin = NULL, limit = 0, currency = NULL,
  offset = NULL)
```

Arguments

`coin`	Token name, default is all, Default: NULL
`limit`	Return top n coins, default is all, Default: 0
`currency`	Convert into local currency. Must be one of 'AUD', 'BRL', 'CAD', 'CHF', 'CLP', 'CNY', 'CZK', 'DKK', 'EUR', 'GBP', 'HKD', 'HUF', 'IDR', 'ILS', 'INR', 'JPY', 'KRW', 'MXN', 'MYR', 'NOK', 'NZD', 'PHP', 'PKR', 'PLN', 'RUB', 'SEK', 'SGD', 'THB', 'TRY', 'TWD', 'ZAR', Default: NULL
`offset`	Optional number to offset the initial index of listings

Details

Updated every 5 minutes

Value

Will provide data frame of current prices

Note

The 'offset' parameter can be used to return the next 100 prices

Examples

```
{
kin_price <- crypto_prices('kin')
}
```

`crypto_prices_full` *Get full listing current crypto currency prices*

Description

This will retrieve the current market prices from CoinMarketCap. Data gets refreshed every 5 minutes and will iterate over all tokens to return the full list of prices.

Usage

```
crypto_prices_full(limit = NULL)
```

Arguments

`limit`	Return top n coins, default is all, Default: 0

Details

Updated every 5 minutes

Value

Will provide data frame of current prices

Examples

```
## Not run:
all_prices <- crypto_prices_full()

## End(Not run)
```

`crypto_timeseries` *Daily crypto currency market data*

Description

Retrieve time series of market_cap, price_btc, price_usd and volume of specified coin - perfect for charting or time series analysis.

Usage

```
crypto_timeseries(coin = NULL)
```

Arguments

`coin`	Name, symbol or slug of crypto currency

Details

Most tokens are refreshed every 6 hours.. results may vary.

Value

Daily time series of token data in a dataframe:

`timestamp`	Timestamp (POSIXct)
`market_cap`	Market Cap in USD
`price_btc`	Price in BTC
`price_usd`	Price in USD
`volume`	Volume traded in USD
`slug`	Coin URL slug (unique)

Examples

```
## Not run:
coin       <- 'kin'
kin_charts <- crypto_timeseries(coin)

## End(Not run)
```

crypto_xts *crypto_xts*

Description

Converts the `getCoins()` dataframe into an xts object. Provide frequency to summarise into specific time periods.

Usage

```
crypto_xts(df, frequency = NULL)
```

Arguments

`df`	data.frame from `getCoins()`
`frequency`	string `?round_date` for help

Value

xts

Note

Each value in `frequency` `<-c('second','minute','hour','day','week','month','year')` can have an integer in front of it to retrieve the expressed time period. i.e. 3month

Examples

```
 ## Not run:
You can lookup additional frequencies at \code{?round_date}
from the lubridate package.
   crypto_xts(df, '.5s')
   crypto_xts(df, 'sec')
   crypto_xts(df, 'second')
   crypto_xts(df, 'minute')
   crypto_xts(df, '5 mins')
   crypto_xts(df, 'hour')
   crypto_xts(df, '2 hours')
   crypto_xts(df, 'day')
   crypto_xts(df, 'week')
   crypto_xts(df, 'month')
   crypto_xts(df, 'bimonth')
   crypto_xts(df, '3 months')
   crypto_xts(df, 'halfyear')
   crypto_xts(df, 'year')

## End(Not run)
```

get_coinlist_api *Get Updated Coinlist*

Description

Gets updated coin listing from CoinMarketCap professional API and prompts user to provide CMC API key using the keyring package.

Usage

```
get_coinlist_api()
```

Value

dataframe

Examples

```
## Not run:
get_coinlist_api()

## End(Not run)
```

get_coinlist_static *Get Updated Coinlist*

Description

Gets updated coin listing from local data

Usage

```
get_coinlist_static()
```

Value

dataframe

Examples

```
## Not run:
get_coinlist_static()

## End(Not run)
```

platform_locale *Platform Locale*

Description

Helper function to identify locale to use for date encoding based on source platform being unix based or windows

Usage

```
platform_locale()
```

Value

locale

Examples

```
{
platform_locale()
}
```

replace_encoding *Check locale encoding*

Description

Helper function to ensure encoding is UTF-8

Usage

```
replace_encoding(sys_locale)
```

Arguments

sys_locale	string system locale

Value

Sets system variable to UTF-8

Examples

```
## Not run:
sys_locale <- Sys.getlocale(category = 'LC_TIME')
replace_encoding(sys_locale)

## End(Not run)
```

reset_encoding *Reset locale encoding*

Description

Helper function to reset encoding from UTF-8 back to R sessions original locale value.

Usage

```
reset_encoding(sys_locale)
```

Arguments

sys_locale	string Original system locale

Value

Sets locale back to original value

Examples

```
## Not run:
sys_locale <- Sys.getlocale(category = 'LC_TIME')
reset_encoding(sys_locale)

## End(Not run)
```

safely_read_json *Safely read json API*

Description

This will attempt to safely retrieve data and return a elegant error message to the user if the data is unavailable.

Usage

```
safely_read_json(json_url)
```

Arguments

json_url	json API URL

Value

parsed dataset or a elegant error message

Examples

```
{
json_url <- 'https://s2.coinmarketcap.com/generated/search/quick_search.json'
result <- safely_read_json(json_url)
}
```

`scraper` *Historical table scraper*

Description

This web scrapes the historic price tables from CoinMarketCap and provides back a dataframe for the coin provided as an input. This function is a dependency of getCoins and is used as part of a loop to retrieve all crypto currencies.

Usage

```
scraper(attributes, slug, sleep = NULL)
```

Arguments

`attributes`	URL generated from `listCoins()`
`slug`	Unique identifier required for merging
`sleep`	Duration to sleep to resolve rate limiter

Value

Raw OHLC market data in a dataframe:

`slug`	Coin url slug
`symbol`	Coin symbol
`name`	Coin name
`date`	Market date
`open`	Market open
`high`	Market high
`low`	Market low
`close`	Market close
`volume`	Volume 24 hours
`market`	USD Market cap

Index