Trade Strategies with Cpp - Ömer Halit Cinar

Generated by Doxygen 1.10.0

Chapter 1

Class Index

1.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

Chapter 2

File Index

2.1 File List

Here is a list of all files with brief descriptions:

Chapter 3

Class Documentation

3.1 Strategy Class Reference

Provides various technical analysis functions for stock prices, such as moving averages, RSI, and Bollinger Bands.

```
#include <strategy.hpp>
```

Static Public Member Functions

- static double calculateShortPeriodMovingAverage (const std::vector< double > &prices)

  Calculates the short-period simple moving average (SMA).
- static double calculateLongPeriodMovingAverage (const std::vector< double > &prices)

  Calculates the long-period simple moving average (SMA).
- static double calculateRSI (const std::vector< double > &prices, int period=14)

  Calculates the Relative Strength Index (RSI) over a specified period.
- static std::tuple< double, double, double > calculateBollingerBands (const std::vector< double > &prices, int period=20, double numStdDev=2.0)

  Calculates the Bollinger Bands for a given set of prices.

Static Public Attributes

- static const int shortPeriod = 5

  Number of periods for short period moving average.
- static const int longPeriod = 20

  Number of periods for long period moving average.

3.1.1 Detailed Description

Provides various technical analysis functions for stock prices, such as moving averages, RSI, and Bollinger Bands.

3.1.2 Member Function Documentation

3.1.2.1 calculateBollingerBands()

```
std::tuple< double, double, double > Strategy::calculateBollingerBands (
             const std::vector< double > & prices,
             int period = 20,
             double numStdDev = 2.0 )  [static]
```

Calculates the Bollinger Bands for a given set of prices.

Parameters

prices	Vector of historical stock prices.
period	Number of periods to use for the moving average (default is 20).
numStdDev	Number of standard deviations to calculate the upper and lower bands (default is 2.0).

Returns

A tuple containing the lower band, middle band (SMA), and upper band.

3.1.2.2 calculateLongPeriodMovingAverage()

```
double Strategy::calculateLongPeriodMovingAverage (
             const std::vector< double > & prices )  [static]
```

Calculates the long-period simple moving average (SMA).

Parameters

prices	Vector of historical stock prices.

Returns

The long-period SMA.

3.1.2.3 calculateRSI()

```
double Strategy::calculateRSI (
             const std::vector< double > & prices,
             int period = 14 )  [static]
```

Calculates the Relative Strength Index (RSI) over a specified period.

Parameters

prices	Vector of historical stock prices.
period	Number of periods to use for RSI calculation (default is 14).

Returns

The RSI value.

3.1.2.4 calculateShortPeriodMovingAverage()

```
double Strategy::calculateShortPeriodMovingAverage (
             const std::vector< double > & prices )  [static]
```

Calculates the short-period simple moving average (SMA).

Parameters

prices	Vector of historical stock prices.

Returns

The short-period SMA.

3.1.3 Member Data Documentation

3.1.3.1 longPeriod

```
const int Strategy::longPeriod = 20  [static]
```

Number of periods for long period moving average.

3.1.3.2 shortPeriod

```
const int Strategy::shortPeriod = 5  [static]
```

Number of periods for short period moving average.

The documentation for this class was generated from the following files:

- inc/strategy.hpp
- src/strategy.cpp

Chapter 4

File Documentation

4.1 inc/strategy.hpp File Reference

```
#include <iostream>
#include <vector>
#include <numeric>
```

Classes

- class Strategy

  Provides various technical analysis functions for stock prices, such as moving averages, RSI, and Bollinger Bands.

4.2 strategy.hpp

Go to the documentation of this file.

```
#ifndef STRATEGY_HPP
#define STRATEGY_HPP

#include <iostream>
#include <vector>
#include <numeric>

class Strategy
{
public:
    const static int shortPeriod = 5;

    const static int longPeriod = 20;

    static double calculateShortPeriodMovingAverage(const std::vector<double>& prices);

    static double calculateLongPeriodMovingAverage(const std::vector<double>& prices);

    static double calculateRSI(const std::vector<double>& prices, int period = 14);

    static std::tuple<double, double, double> calculateBollingerBands(const std::vector<double>&
prices, int period = 20, double numStdDev = 2.0);
};

#endif // STRATEGY_HPP
```

4.3 inc/utils.hpp File Reference

```
#include <string>
#include <vector>
```

Functions

- void loadEnv ()

  Loads environment variables from the `.env` file.
- std::string fetchMarketData (const std::string &apiKey, const std::string &symbol)

  Fetches market data for a given stock symbol using an API.
- std::string fetchAvailableStocks (const std::string &apiKey)

  Fetches available stocks from the API.
- std::vector< double > parseMarketData (const std::string &jsonData)

  Parses the market data JSON string and extracts close prices.
- void movingAverageLogger (const double shortSMA, const double longSMA)

  Logs buy, sell, or hold signals based on short and long simple moving averages (SMA).
- void rsiLogger (const double rsi)

  Logs buy, sell, or hold signals based on the Relative Strength Index (RSI).
- void bollingerBandsLogger (const double lowerBand, const double middleBand, const double upperBand, const double lastPrice)

  Logs buy, sell, or hold signals based on Bollinger Bands.
- void loadMenu ()

  Loads the menu options for the application.
- std::vector< std::string > getAvailableStocks ()

  Fetches the list of available stocks.
- std::vector< std::string > parseAvailableStocks (const std::string &jsonData)

  Parses the JSON response and extracts available stock symbols.

4.3.1 Function Documentation

4.3.1.1 bollingerBandsLogger()

```
void bollingerBandsLogger (
            const double lowerBand,
            const double middleBand,
            const double upperBand,
            const double lastPrice )
```

Logs buy, sell, or hold signals based on Bollinger Bands.

Parameters

lowerBand	The lower Bollinger Band.
middleBand	The middle Bollinger Band (SMA).
upperBand	The upper Bollinger Band.
lastPrice	The most recent stock price.

4.3.1.2 fetchAvailableStocks()

```
std::string fetchAvailableStocks (
            const std::string & apiKey )
```

Fetches available stocks from the API.

Parameters

apiKey	The API key for authentication.

Returns

The available stocks as a JSON string.

4.3.1.3 fetchMarketData()

```
std::string fetchMarketData (
            const std::string & apiKey,
            const std::string & symbol )
```

Fetches market data for a given stock symbol using an API.

Parameters

apiKey	The API key for authentication.
symbol	The stock symbol to fetch data for.

Returns

The market data as a JSON string.

4.3.1.4 getAvailableStocks()

```
std::vector< std::string > getAvailableStocks ( )
```

Fetches the list of available stocks.

Returns

A vector containing stock symbols.

4.3.1.5 loadEnv()

```
void loadEnv ( )
```

Loads environment variables from the `.env` file.

4.3.1.6 loadMenu()

```
void loadMenu ( )
```

Loads the menu options for the application.

4.3.1.7 movingAverageLogger()

```
void movingAverageLogger (
            const double shortSMA,
            const double longSMA )
```

Logs buy, sell, or hold signals based on short and long simple moving averages (SMA).

Parameters

shortSMA	The short-period SMA.
longSMA	The long-period SMA.

4.3.1.8 parseAvailableStocks()

```
std::vector< std::string > parseAvailableStocks (
            const std::string & jsonData )
```

Parses the JSON response and extracts available stock symbols.

Parameters

jsonData	The JSON data containing stock information.

Returns

A vector of stock symbols.

4.3.1.9 parseMarketData()

```
std::vector< double > parseMarketData (
            const std::string & jsonData )
```

Parses the market data JSON string and extracts close prices.

Parameters

jsonData	The JSON data containing stock prices.

Returns

A vector of close prices.

4.3.1.10 rsiLogger()

```
void rsiLogger (
            const double rsi )
```

Logs buy, sell, or hold signals based on the Relative Strength Index (RSI).

Parameters

rsi	The RSI value.

4.4 utils.hpp

Go to the documentation of this file.

```
#ifndef UTILS_HPP
#define UTILS_HPP

#include <string>
#include <vector>

void loadEnv();

std::string fetchMarketData(const std::string& apiKey, const std::string& symbol);

std::string fetchAvailableStocks(const std::string& apiKey);

std::vector<double> parseMarketData(const std::string& jsonData);

void movingAverageLogger(const double shortSMA, const double longSMA);

void rsiLogger(const double rsi);

void bollingerBandsLogger(const double lowerBand, const double middleBand, const double upperBand,
      const double lastPrice);

void loadMenu();

std::vector<std::string> getAvailableStocks();

std::vector<std::string> parseAvailableStocks(const std::string& jsonData);

#endif // UTILS_HPP
```

4.5 src/main.cpp File Reference

```
#include <iostream>
#include <string>
#include <vector>
#include "../inc/utils.hpp"
#include "../inc/strategy.hpp"
```

Functions

- int main ()

4.5.1 Function Documentation

4.5.1.1 main()

```
int main ( )
```

4.6 src/strategy.cpp File Reference

```
#include "../inc/strategy.hpp"
#include <cmath>
#include <numeric>
```

4.7 src/utils.cpp File Reference

```
#include "../inc/utils.hpp"
#include "../inc/strategy.hpp"
#include <iostream>
#include <fstream>
#include <sstream>
#include <curl/curl.h>
#include "../json/json.hpp"
```

Functions

- size_t WriteCallback (void *contents, size_t size, size_t nmemb, void *userp)
- void loadEnv ()

    Loads environment variables from the `.env` file.
- std::string fetchMarketData (const std::string &apiKey, const std::string &symbol)

    Fetches market data for a given stock symbol using an API.
- std::string fetchAvailableStocks (const std::string &apiKey)

    Fetches available stocks from the API.
- std::vector< double > parseMarketData (const std::string &jsonData)

    Parses the market data JSON string and extracts close prices.
- std::vector< std::string > parseAvailableStocks (const std::string &jsonData)

    Parses the JSON response and extracts available stock symbols.
- std::vector< std::string > getAvailableStocks ()

    Fetches the list of available stocks.
- void movingAverageLogger (const double shortSMA, const double longSMA)

    Logs buy, sell, or hold signals based on short and long simple moving averages (SMA).
- void rsiLogger (const double rsi)

    Logs buy, sell, or hold signals based on the Relative Strength Index (RSI).
- void bollingerBandsLogger (const double lowerBand, const double middleBand, const double upperBand, const double lastPrice)

    Logs buy, sell, or hold signals based on Bollinger Bands.
- void loadMenu ()

    Loads the menu options for the application.

4.7.1 Function Documentation

4.7.1.1 bollingerBandsLogger()

```
void bollingerBandsLogger (
            const double lowerBand,
            const double middleBand,
            const double upperBand,
            const double lastPrice )
```

Logs buy, sell, or hold signals based on Bollinger Bands.

Parameters

lowerBand	The lower Bollinger Band.
middleBand	The middle Bollinger Band (SMA).
upperBand	The upper Bollinger Band.
lastPrice	The most recent stock price.

4.7.1.2 fetchAvailableStocks()

```
std::string fetchAvailableStocks (
            const std::string & apiKey )
```

Fetches available stocks from the API.

Parameters

apiKey	The API key for authentication.

Returns

The available stocks as a JSON string.

4.7.1.3 fetchMarketData()

```
std::string fetchMarketData (
            const std::string & apiKey,
            const std::string & symbol )
```

Fetches market data for a given stock symbol using an API.

Parameters

apiKey	The API key for authentication.
symbol	The stock symbol to fetch data for.

Returns

The market data as a JSON string.

4.7.1.4 getAvailableStocks()

```
std::vector< std::string > getAvailableStocks ( )
```

Fetches the list of available stocks.

Returns

A vector containing stock symbols.

4.7.1.5 loadEnv()

```
void loadEnv ( )
```

Loads environment variables from the `.env` file.

4.7.1.6 loadMenu()

```
void loadMenu ( )
```

Loads the menu options for the application.

4.7.1.7 movingAverageLogger()

```
void movingAverageLogger (
             const double shortSMA,
             const double longSMA )
```

Logs buy, sell, or hold signals based on short and long simple moving averages (SMA).

Parameters

shortSMA	The short-period SMA.
longSMA	The long-period SMA.

4.7.1.8 parseAvailableStocks()

```
std::vector< std::string > parseAvailableStocks (
             const std::string & jsonData )
```

Parses the JSON response and extracts available stock symbols.

Parameters

jsonData	The JSON data containing stock information.

Returns

A vector of stock symbols.

4.7.1.9 parseMarketData()

```
std::vector< double > parseMarketData (
            const std::string & jsonData )
```

Parses the market data JSON string and extracts close prices.

Parameters

jsonData	The JSON data containing stock prices.

Returns

A vector of close prices.

4.7.1.10 rsiLogger()

```
void rsiLogger (
            const double rsi )
```

Logs buy, sell, or hold signals based on the Relative Strength Index (RSI).

Parameters

rsi	The RSI value.

4.7.1.11 WriteCallback()

```
size_t WriteCallback (
            void * contents,
            size_t size,
            size_t nmemb,
            void * userp )
```