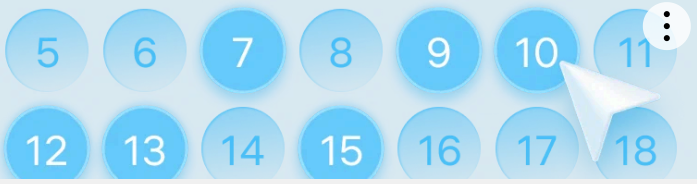



empenoso
8 ноя 2024 в 08:33

# Инструменты робота, торгующего на Московской бирже через API брокера

Сложный 8 мин 1.8K

Open source*, Финансы в IT, JavaScript*, Node.JS*

Кейс

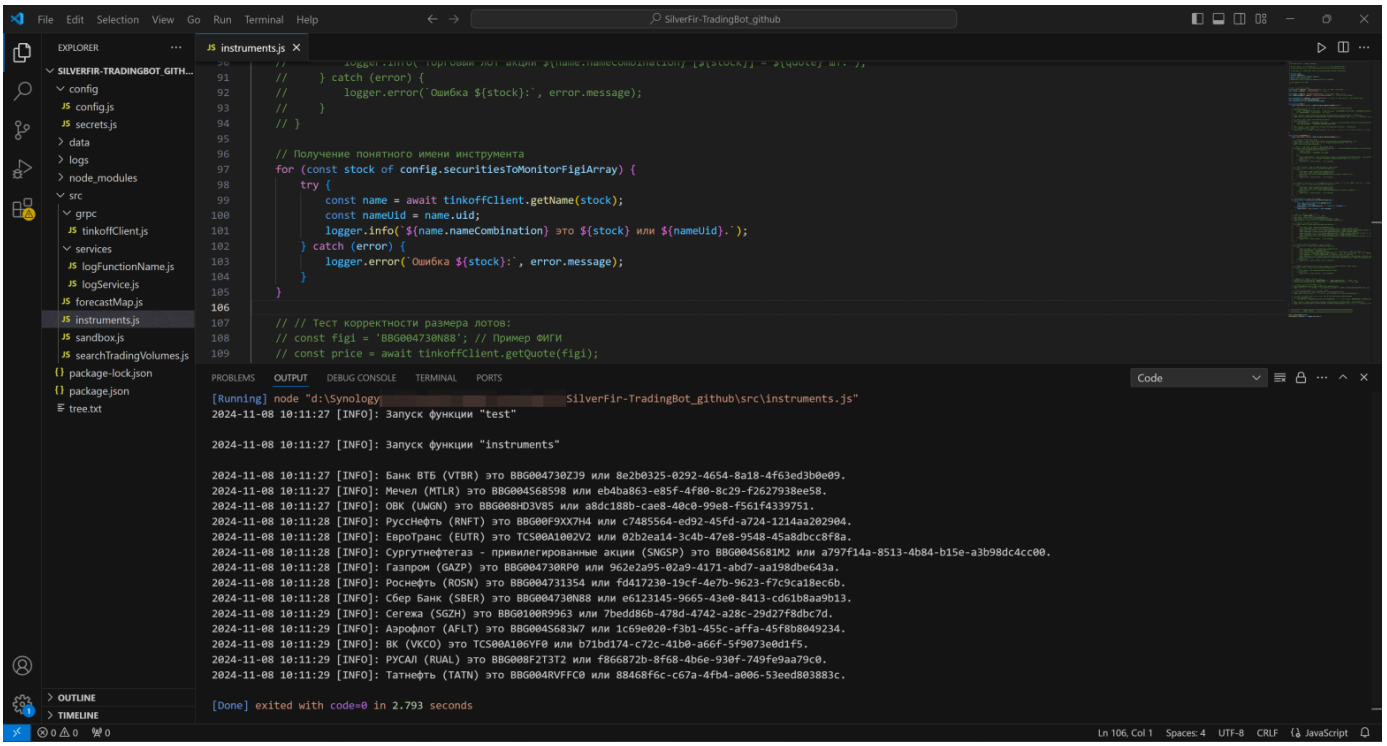
Поскольку хочу использовать для среднесрочной алгоритмической торговли на российском рынке скрипт - робота, то мне необходимо получать от брокера актуальную информацию о текущих ценах и сопутствующую информацию:

- Время работы биржи через `InstrumentsService/TradingSchedules`.
- Основную информацию об инструменте через `InstrumentsService/GetInstrumentBy`.
- Последнюю котировку по инструменту через `MarketDataService/GetLastPrices`.
- Торговые лоты - это определенное количество акций, которые можно купить или продать в рамках одной сделки.
- Свечи по инструменту для разных временных интервалов через `MarketDataService/GetCandles`.
- Технические индикаторы через `MarketDataService/GetTechAnalysis`.
- Понятное имя инструмента через `InstrumentsService/FindInstrument`.

В статье разбираюсь как проделать все эти операции при помощи программного кода.

Частному лицу для начала торговли на бирже частному инвестору необходим брокерский счёт. Но лишь у немногих российских брокеров есть собственные API (точно есть у ФИНАМ, Алор, Тинькофф Инвестиции). По личным предпочтениям я решил использовать

API от Т-Банк (ранее известный как Тинькофф), работая в среде исполнения JavaScript Node.js.




Время запроса почти 3 секунды - это много

## SilverFir-TradingBot\src\instruments.js

Этот модуль служит для проверки части функций, которые будут использоваться потом в автоматическом режиме. Что он делает? Импортирует необходимые модули:

- `secrets` и `config` для конфиденциальной информации и настроек конфигурации.
- Службы для рисования диаграмм ( `chart` ), обработки CSV-файлов ( `csvHandler` ), решений о покупке/продаже ( `buyDecision` и `sellDecision` ) и расчета доходности ( `yieldCalculator` ).
- Служба ведения журнала ( `logger` ) для отслеживания действий и ошибок.
- `TinkoffClient` , модуль для взаимодействия с Tinkoff Invest API, и `API_TOKEN` для аутентификации.

### Основные функции

Функция `test()` :

**Цель**: Тестирование функциональности API и регистрация данных для конкретных биржевых инструментов.

**Примеры операций**:

- Получить основную информацию об инструменте - вызывает `InstrumentsService/GetInstrumentBy` для получения информации о определенном инструменте с использованием его идентификатора.
- Получить список всех акций - вызывает `InstrumentsService/Shares` для составления списка акций и регистрации первых нескольких результатов.

Функция `instruments()` :

**Цель**: Основная функция для извлечения данных и подготовки к торговле.

**Примеры операций**:

- Получение времени работы биржи - получает и регистрирует часы торговли.
- Найти всю информацию об акциях в списке файла config - отображает всю информацию о каждом из тикеров в JSON формате.
- Последние цены и торговые лоты - извлекает последние цены акций и проверяет размеры лотов (это определенное количество акций, которые можно купить или продать в рамках одной сделки).
- Данные свечей - собирает данные свечей (ценовые точки с течением времени) в различные интервалы (5 минут, час, день).
- Технические индикаторы - извлекает индикаторы, такие как SMA (простая скользящая средняя), для анализа тенденций акций. По выходным данных нет, хотя свечи за это же время присутствуют.
- Разместить рыночный ордер - строки кода для прямого размещения ордеров на покупку/продажу.
- Позиции портфеля - перечисляет текущие активы и вычисляет годовую доходность.

В конце код запускает `test()` и `instruments()` с обработкой ошибок, регистрируя все возникшие проблемы.

Файл `instruments.js` это ещё одна часть бота, которая позволяет частному инвестору отслеживать и взаимодействовать с платформой Tinkoff, обрабатывая все: от анализа цен

акций и тенденций до размещения сделок. Настройка этого бота подходит для среднесрочной торговли на основе данных, используя Node.js для быстрой обработки данных и взаимодействия с API.

```
// Импорт необходимых модулей
const secrets = require('../config/secrets'); // Ключи доступа и идентификаторы
const config = require('../config/config'); // Параметры
const chart = require('./services/chartService'); // Отрисовка графиков
const csvHandler = require('./services/csvHandler'); // Работа с CSV файлами
const buyDecision = require('./services/buyDecision'); // Функции покупки
const sellDecision = require('./services/sellDecision'); // Функции продажи
const yieldCalculator = require('./services/yieldCalculator'); // Расчёт годовой доходн

const logger = require('./services/logService'); // Логирование в файл и консоль
const logFunctionName = require('./services/logFunctionName'); // Получение имени функц

const TinkoffClient = require('./grpc/tinkoffClient'); // модуль для взаимодействия с A
const API_TOKEN = secrets.TbankSandboxMode;
const tinkoffClient = new TinkoffClient(API_TOKEN);

async function test() {
    logger.info(`Запуск функции ${JSON.stringify(logFunctionName())}\n`);

    // // Получить основную информацию об инструменте InstrumentsService/GetInstrumentB
    // const testPayload = {
    //     idType: "INSTRUMENT_ID_TYPE_FIGI", // Тип идентификатора INSTRUMENT_ID_TYPE_
    //     id: "BBG004730N88" // Идентификатор инструмента
    // };
    // const response = await tinkoffClient.callApi('InstrumentsService/GetInstrumentBy
    // logger.info(`InstrumentsService/GetForecastBy: ${JSON.stringify(response, null,

    // // Получить список акций InstrumentsService/Shares
    // const testPayload = {
    //     "instrumentStatus": "INSTRUMENT_STATUS_BASE", // https://russianinvestments.
    //     "instrumentExchange": "INSTRUMENT_EXCHANGE_UNSPECIFIED"
    // };
    // const response = await tinkoffClient.callApi('InstrumentsService/Shares', testPa
    // // Отображение ответа от API
    // logger.info(`Ответ: ${JSON.stringify(response, null, 2)}`); // выводится только

}
```

```
async function instruments() {
    logger.info(`Запуск функции ${JSON.stringify(logFunctionName())}\n`);

    // // Получение времени работы биржи
    // const response = await tinkoffClient.callApi('InstrumentsService/TradingSchedule
    // logger.info(`Получение времени работы биржи: ${JSON.stringify(response, null, 2)
    // await tinkoffClient.getExchangeOpen();

    // // Найти всю информацию об акциях в списке файла config
    // for (const stock of config.securitiesToMonitorTikerArray) { // securitiesToMonit
    //     const securitiesToMonitorTikerArrayPayload = {
    //         "query": stock,
    //         "instrumentKind": "INSTRUMENT_TYPE_SHARE"
    //     };
    //     try {
    //         const FindInstrument = await tinkoffClient.callApi('InstrumentsService/F
    //         logger.info(`Ищем тикер ${stock}:\n${JSON.stringify(FindInstrument, null
    //     } catch (error) {
    //         logger.error(`Ошибка ${stock}:`, error.message);
    //     }
    // }

    // // Получить последнюю цену для акций из списка в файле config
    // for (const stock of config.securitiesToMonitorFigiArray) {
    //     try {
    //         const quote = await tinkoffClient.getQuote(stock);
    //         const name = await tinkoffClient.getName(stock);
    //         logger.info(`Цена акции ${name.nameCombination} [${stock}]: ${quote} руб
    //     } catch (error) {
    //         logger.error(`Ошибка ${stock}:`, error.message);
    //     }
    // }

    // // Получение торговых лотов - это определенное количество акций, которые можно к
    // for (const stock of config.securitiesToMonitorFigiArray) {
    //     try {
    //         const quote = await tinkoffClient.getLot(stock);
    //         const name = await tinkoffClient.getName(stock);
    //         logger.info(`Торговый лот акции ${name.nameCombination} [${stock}] = ${q
    //     } catch (error) {
    //         logger.error(`Ошибка ${stock}:`, error.message);
    //     }
    // }
```

```
// Получение понятного имени инструмента
for (const stock of config.securitiesToMonitorFigiArray) {
    try {
        const name = await tinkoffClient.getName(stock);
        const nameUid = name.uid;
        logger.info(`${name.nameCombination} это ${stock} или ${nameUid}.`);
    } catch (error) {
        logger.error(`Ошибка ${stock}:`, error.message);
    }
}

// // Тест корректности размера лотов:
// const figi = 'BBG004730N88'; // Пример ФИГИ
// const price = await tinkoffClient.getQuote(figi);
// const quantity = await config.getPurchaseQuantity(price, figi);
// logger.info(`Тест количества лотов ${figi} для покупки: ${quantity}`);

// // Получение свечей по инструменту
// for (const stock of config.securitiesToMonitorFigiArray) {
//     try {
//         const name = await tinkoffClient.getName(stock);
//         const candles5Min = await tinkoffClient.getCandles(stock, "CANDLE_INTERV
//         logger.info(`5-минутные свечи для ${name.nameCombination}: ${JSON.string
//         const candlesHour = await tinkoffClient.getCandles(stock, "CANDLE_INTERV
//         logger.info(`Часовые свечи для ${name.nameCombination}: ${JSON.stringify
//         const candlesDay = await tinkoffClient.getCandles(stock, "CANDLE_INTERVA
//         logger.info(`Дневные свечи для ${name.nameCombination}: ${JSON.stringify
//     } catch (error) {
//         logger.error(`Ошибка ${stock}:`, error.message);
//     }
// }

// // Получение технических индикаторов по инструменту
// for (const stock of config.securitiesToMonitorFigiArray) {
//     try {
//         const instrument = await tinkoffClient.getName(stock);
//         const instrumentUid = instrument.uid;
//         const indicatorType = "INDICATOR_TYPE_SMA"; // Пример типа индикатора (S
//         const interval = "INDICATOR_INTERVAL_FIVE_MINUTES"; // Пример интервала
//         const typeOfPrice = "TYPE_OF_PRICE_CLOSE"; // Тип цены (например, закрыт
//         const indicators = await tinkoffClient.getTechIndicators(instrumentUid,
//         logger.info(`Индикатор ${indicatorType} для ${instrument.nameCombination
```

```
    //     } catch (error) {
    //         logger.error(`Ошибка ${stock}: ${error.message}`);
    //     }
    // }

    // // Создание графиков пересечения свечей и индикатора для акций из списка в файле
    // for (const stock of config.securitiesToMonitorFigiArray) {
    //     try {
    //         const charts = chart.generateCandlestickChart(stock);
    //     } catch (error) {
    //         logger.error(`Ошибка ${stock}:`, error.message);
    //     }
    // }

    // // Функция для отправки рыночного ордера
    // tinkoffClient.placeMarketOrder('BBG004730N88', 1, 'ORDER_DIRECTION_BUY'); // Куп
    // tinkoffClient.placeMarketOrder('BBG004730N88', 1, 'ORDER_DIRECTION_SELL'); // Пр

    // // Получить все открытые позиции счёта
    // const GetSandboxPositions = await tinkoffClient.getPortfolio();
    // logger.info(`Все открытые позиции счёта ${secrets.AccountID}:\n ${JSON.stringify

    // // Расчёт годовой доходности от Торгового робота
    // const SilverFirBotYield = await yieldCalculator.calculateAnnualYield();
    // logger.info(`Годовая доходность от Торгового робота SilverFir Bot: ${SilverFirBo

    // // Получить прогнозов инвестдомов по инструменту InstrumentsService/GetForecastB
    // const ForecastPayload = {
    //     "instrumentId": "1c69e020-f3b1-455c-affa-45f8b8049234" // У Аэрофлот (AFLT),
    // };
    // const response = await tinkoffClient.callApi('InstrumentsService/GetForecastBy',
    // logger.info(`InstrumentsService/GetForecastBy: ${JSON.stringify(response, null,
}


// ================================================================================
// ============      Запуск функций   =============================================
// ================================================================================

test().catch(logger.error);
instruments().catch(err => logger.error(err));
```

Многие строки закомментированы, но это не потому что они не рабочие, а потому что они используются для тестов той или иной функции.

**Итоги**

Проект полностью представлен на Гитхабе: https://github.com/empenoso/SilverFir-TradingBot. Новые модули будут загружаться по мере написания и тестирования.

Автор: Михаил Шардин

11 ноября 2024 г.

**Теги:** бот, московская биржа, мосбиржа, moexalgo, tbank, T-Bank Invest API

**Хабы:** Open source, Финансы в IT, JavaScript, Node.JS



## Публикации

ЛУЧШИЕ ЗА СУТКИ ПОХОЖИЕ

**xjr358**
23 часа назад

## IBM 600E Ретро ноутбук за 2000 тенге (400 рублей)

Простой 3 мин 5.6K

Обзор

+37 12 39

**Erwinmal**
5 часов назад

## Кто поджёг Лос-Анджелес? Свежая конспирология о виноватых НЛО, Пи Дидди, урбанистах и корюшке

Простой 14 мин 3.2K

Обзор

+26 4 31

**DimDimDimDimDim**
6 часов назад

## Rust 1.84: новый релиз отличного языка программирования. Еще лучше, еще эффективнее, как всегда

6 мин 2.1K

+17 8 2

**JBFW**
14 часов назад

## Подключаем длинную линию 1-wire к Ардуино

3 мин 4.1K

+17 32 27

**DENEVGTAR**
4 часа назад

## Распознавание образов в мозге с помощью микроплееров

Средний 8 мин 897

Из песочницы

+11 15 5

**chlorine**
6 часов назад

## Кэш. Теория кэширования. Устройство и разновидности кэша

Простой 7 мин 1.8K

Из песочницы

+11 60 16

**mikhailmurzak**
20 часов назад

## Делаем Телеграм-бота в Cursor AI без знания кода

Простой 5 мин 6.4K

Туториал

+11 73 15

**arturdumchev**
46 минут назад

## Заговор разработчиков против корпораций

Средний 15 мин 786

Мнение

+10 2 0

**burenkov**
2 часа назад

## Стереокамера машинного зрения с поддержкой ИИ на базе FPGA и Arduino Portenta H7

10 мин 583

Из песочницы

+10 10 0

avovana7
23 часа назад

## System Design для начинающих: всё, что вам нужно. Часть 1

Простой 13 мин 7.9K

Туториал Recovery Mode Перевод

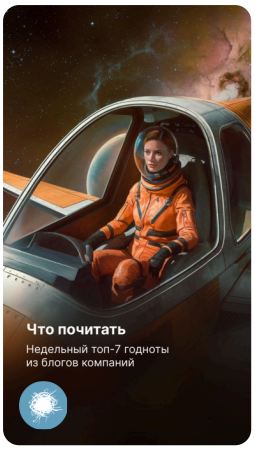
+10 195 7

## Обзор наушников HUAWEI FreeBuds Pro 4: оцениваем звук, шумодав и время автономной работы

Турбо

Показать еще

ИСТОРИИ



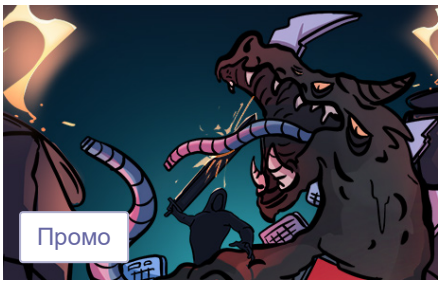

**Годнота из блогов компаний**


**Выравнивания планет**

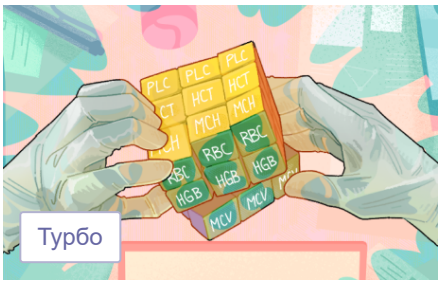

**Нейрозима 2025**


**Статьи с новогодним вайбом**


**Кто выступит на конференции мечты**




**Чтобы победить Переработу, нужно всего лишь...**




**Один анализ крови, чтобы править всеми**


**Как хабравчане следят за здоровьем?**

## РАБОТА

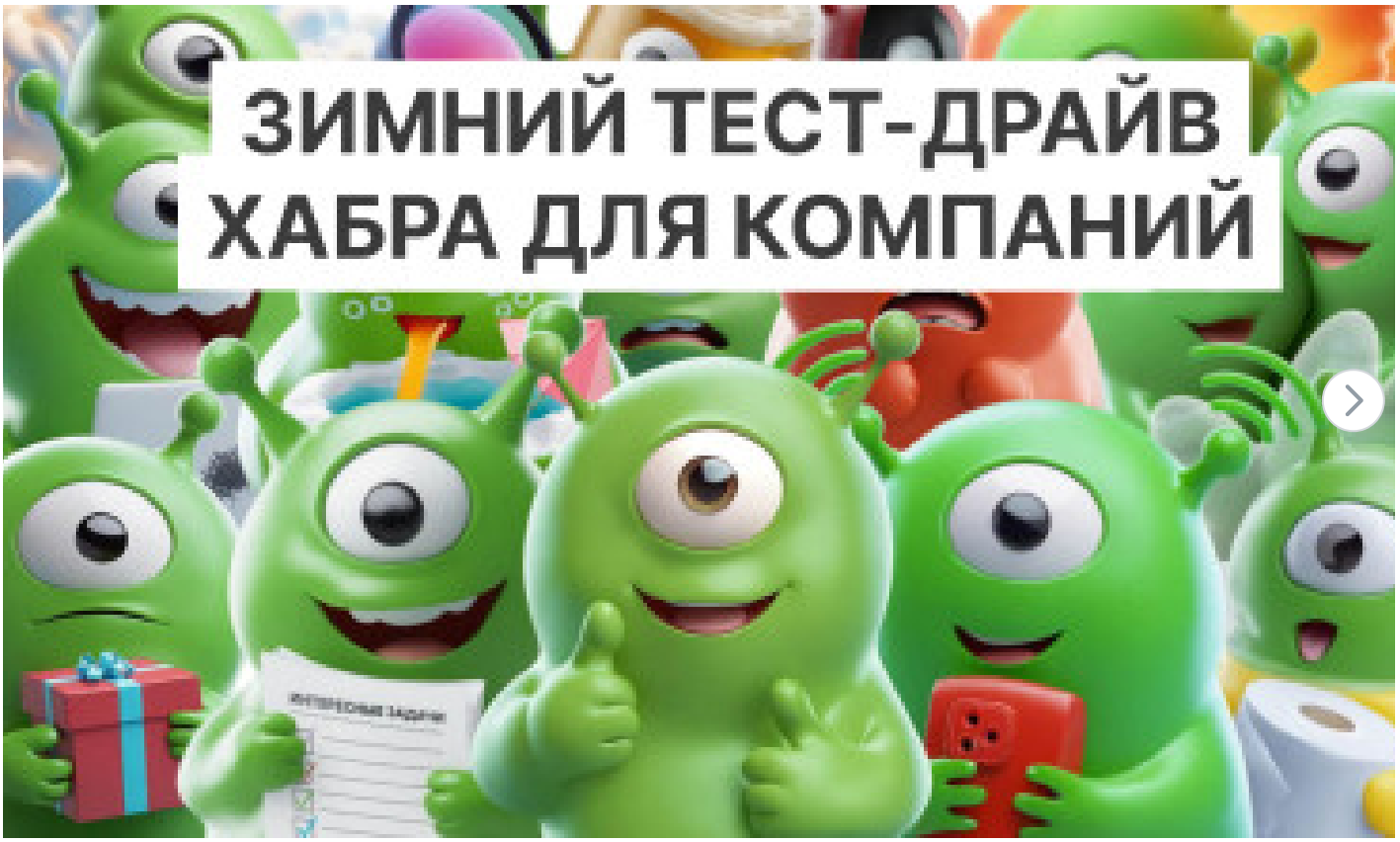
React разработчик
30 вакансий

Node.js разработчик
33 вакансии

JavaScript разработчик
98 вакансий

Все вакансии

## БЛИЖАЙШИЕ СОБЫТИЯ




30 января

## Зимний тест-драйв Хабра для компаний

Москва

Маркетинг Другое

Больше событий в календаре