

Хабр



КАК СТАТЬ АВТОРОМ



Зарплаты айтишников

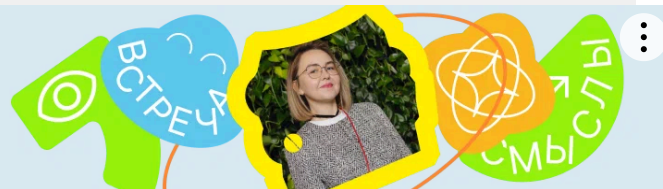


Войти

РЕКЛАМА

Хабр Семинар

Как PR продукта формирует
сильный IT-бренд



empenoso

27 ноя 2024 в 03:23

Отслеживание позиций торгового робота Московской биржи через CSV файл



Простой



7 мин



2.2K

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

Кейс

Нахожусь в процессе написания механизма торгового робота, работающего на Московской бирже через API одного из брокеров. Брокеров имеющих своё АПИ для МосБиржи катастрофически мало — мне известно только о трёх. При этом, когда я стал публиковать модули робота (и полностью [выложу готовый механизм робота на GitHub](#)), то стал получать непонимание — например, мне писали в комментариях — зачем придумывать велосипед, когда уже есть QUIK — популярная российская платформа для биржевых торгов. В Квике уже есть готовый функционал «импорт транзакций из файла» или таблица «карман транзакций». В тех же комментариях предлагали даже рассмотреть использование платформы 1С для робота, но оказалось, что торговля все равно будет осуществляться через импорт `.tri`-файла в Квик.

Лично мне Квик не очень нравится тем, что это программа для Windows. Хочется иметь механизм торгового робота, который был бы кроссплатформенным и легким — это позволит использовать его даже на «слабом» сервере. К тому же, [много лет назад](#), когда Квик был единственной альтернативой для частного лица, невозможно было внутри одной

**Культура в компании**

Что вы о ней думаете?

отправляла свои сигналы на покупку и продажу в соответствующий Квик. Это было нужно для разных торговых стратегий.

По субъективным причинам я стал писать торгового робота в среде исполнения JavaScript Node.js, но для тестирования на истории пришлось использовать Python и его библиотеки.

```

src > services > JS csvHandler.js > on('data') callback
19 const path = require('path'); // модуль для работы с путями файлов и директорий
20 const filePath = path.join(__dirname, '../data/positions.csv'); // Путь к файлу CSV
21 const logger = require('./logService'); // Подключаем модуль для логирования
22 const logFunctionName = require('./logFunctionName'); // Модуль для получения имени функции (для логирования)
23
24
25 // Загружаем все позиции из CSV файла
26 function loadPositions() {
27   return new Promise((resolve, reject) => {
28     const positions = [];
29     fs.createReadStream(filePath)
30       .pipe(csv())
31       .on('data', (row) => {
32         positions.push({
33           ticker: row.ticker,
34           figi: row.figi,
35           quantity: parseFloat(row.quantity), // Преобразование количества в float
36           purchaseDate: row.purchaseDate,
37           purchasePrice: parseFloat(row.purchasePrice), // Преобразование цены покупки в float
38           updateDate: row.updateDate,
39           maxPrice: parseFloat(row.maxPrice), // Преобразование максимальной цены в float
40           profitLoss: parseFloat(row.profitLoss) // Преобразование прибыли/убытков в float
41         });
42       })
43       .on('end', () => resolve(positions))
44       .on('error', reject);
45   });
46 }
47
48 // Сохраняем актуальные данные о позициях в CSV файл
49 function savePositions(positions) {
50   const csvFields = ['ticker', 'figi', 'quantity', 'purchaseDate', 'purchasePrice', 'updateDate', 'maxPrice', 'profitLoss'];
51   const csvData = parse(positions, { fields: csvFields });
52   fs.writeFileSync(filePath, csvData);
53 }
  
```

Модуль, считывающий позиции из файла

Проблемы с записью позиций в Node.js

Вообще именно этот модуль пришлось пару раз переписывать, потому что не смог сразу отладить его. Проблема была в том, что вызов модуля записи и обновления позиций осуществлялся сразу из нескольких мест и одни результаты перезаписывали другие. Но удалось разобраться и теперь всё протестировано и работает.

Дополнительно использую библиотеки `csv-parser` и `json2csv` — это популярные инструменты Node.js для обработки данных CSV, каждая из которых служит различным целям:

- **csv-parser**, это легкая и быстрая библиотека для анализа файлов CSV. Она основана на потоках, что делает ее очень эффективной для обработки больших наборов данных



Культура в компании

Что вы о ней думаете?

работать как синхронно, так и асинхронно.

Установка этих библиотек:

```
npm install csv-parser json2csv
```

Мой модуль csvHandler.js

Этот код определяет модуль для взаимодействия с CSV-файлом для управления финансовыми торговыми позициями. Служит для загрузки, сохранения, обновления и удаления финансовых позиций, хранящихся в CSV-файле.

Ключевые библиотеки:

- `fs` : для операций файловой системы, таких как чтение и запись файлов.
- `csv-parser` : для анализа CSV-файлов в объекты JavaScript.
- `json2csv` : для преобразования объектов JavaScript в формат CSV для сохранения.
- `path` : для управления путями к файлам.
- **Интеграция**: включает пользовательские модули для ведения журнала (`logService`) и получения имен функций для лучшей отладки.

Функциональность

1. Обработка пути к файлу: использует модуль `path` для поиска CSV-файла, хранящего данные о позиции: `../../data/+positions.csv` .

2. Функции управления позицией:

`loadPositions()`:

1. Считывает CSV-файл и анализирует его в массив объектов позиции.

2. Преобразует числовые поля (`quantity`, `purchasePrice`, `maxPrice`, `profitLoss`) в числа с плавающей точкой для вычислений.



Культура в компании

Что вы о ней думаете?

savePositions(positions):

1. Преобразует массив объектов позиции обратно в формат CSV с помощью json2csv.
2. Перезаписывает CSV-файл обновленными данными.

removePosition(figi):

1. Удаляет позицию из CSV-файла на основе ее figi (уникального идентификатора).
2. Загружает все позиции, отфильтровывает указанную и перезаписывает файл.

updatePosition(newPosition):

1. Добавляет новую позицию или обновляет существующую в CSV-файле:
 2. Если figi существует, обновляет соответствующую позицию.
 3. В противном случае добавляет новую позицию.
 4. Сохраняет обновленный список обратно в CSV-файл.
3. Экспортированные модули: функции `loadPositions`, `updatePosition` и `removePosition` для использования в других частях робота.

Полный код csvHandler.js:

```
const fs = require('fs');
const csv = require('csv-parser');
const { parse } = require('json2csv');
const path = require('path'); // Модуль для работы с путями файлов и директорий
const filePath = path.join(__dirname, '../data/positions.csv'); // Путь к файлу CSV
const logger = require('./logService'); // Подключаем модуль для логирования
const logFunctionName = require('./logFunctionName'); // Модуль для получения имени фун
```

```
// Загружаем все позиции из CSV файла
```

```
function loadPositions() {
  return new Promise((resolve, reject) => {
```

**Культура в компании**

Что вы о ней думаете?

```
.pipe(csv())
```

```

        .on('data', (row) => {
            positions.push({
                ticker: row.ticker,
                figi: row.figi,
                quantity: parseFloat(row.quantity), // Преобразование количества в
                purchaseDate: row.purchaseDate,
                purchasePrice: parseFloat(row.purchasePrice), // Преобразование цен
                updateDate: row.updateDate,
                maxPrice: parseFloat(row.maxPrice), // Преобразование максимальной
                profitLoss: parseFloat(row.profitLoss) // Преобразование прибыли/уб
            });
        })
        .on('end', () => resolve(positions))
        .on('error', reject);
    });
}

// Сохраняем актуальные данные о позициях в CSV файл
function savePositions(positions) {
    const csvFields = ['ticker', 'figi', 'quantity', 'purchaseDate', 'purchasePrice', '
    const csvData = parse(positions, { fields: csvFields });

    fs.writeFileSync(filePath, csvData);
}

// Удаляем позицию из CSV файла (после продажи)
function removePosition(figi) {
    loadPositions().then(positions => {
        const updatedPositions = positions.filter(position => position.figi !== figi);
        savePositions(updatedPositions);
    });
}

// Добавляем новую позицию или обновляем существующую в CSV файле
function updatePosition(newPosition) {
    loadPositions().then(positions => {
        const index = positions.findIndex(pos => pos.figi === newPosition.figi);

        if (index === -1) {
            // Добавляем, если не нашли существующую позицию

```



Культура в компании

Что вы о ней думаете?

```
        positions[index] = newPosition;
    }

    savePositions(positions);
});
}

module.exports = { loadPositions, updatePosition, removePosition };
```

Мой модуль checkCSVpositions.js

Этот модуль важен для обеспечения согласованности данных между локальным CSV-файлом и текущими позициями, полученными из T-Bank Invest API. Он проверяет наличие несоответствий, которые могут привести к ошибкам в торговых операциях, и останавливает робота, если обнаруживаются несоответствия.

Основные функции

1. Интеграция с внешними системами

- T-Bank Invest API: взаимодействует с API для извлечения торговых позиций в реальном времени.
- CSV File Management: использует локальный CSV-файл для хранения и управления представлением бота о торговых позициях.

2. Проверка согласованности

- Сравнивает позиции из CSV-файла с позициями с сервера T-Bank Invest API.
- Проверяет как количество, так и наличие позиций для обнаружения несоответствий.

3. Обработка ошибок

- Регистрирует подробные ошибки при обнаружении несоответствий.
- Останавливает торговые операции для предотвращения дальнейших действий на



Культура в компании

Что вы о ней думаете?

Основные функции:

1. `getServerPositions()`

- Извлекает все открытые позиции из T-Bank Invest API.
- Извлекает позиции с ценными бумагами и преобразует баланс в float для сравнения.
- Регистрирует ответ сервера для отладки и аудита.

2. `checkForDiscrepancies()`

- Загружает данные CSV: считывает локальную запись позиций бота с помощью `csvHandler`.

Сравнивает позиции:

- Для каждой позиции CSV ищет соответствующую позицию на сервере с помощью FIGI (уникальный идентификатор).
- Извлекает размер лота для точного сравнения количества.
- Если обнаружены расхождения в количестве или отсутствующие позиции, регистрирует ошибки и останавливает торговлю.
- Статус журнала: подтверждает, когда все позиции совпадают, и позволяет продолжить торговлю.

Рабочий процесс

1. Извлечение позиций:

- Локальные позиции загружаются из CSV-файла.
- Позиции сервера извлекаются через API Tinkoff.

2. Обнаружение расхождений:

Для каждой позиции в CSV-файле:

3. Код вычисляет общее количество в лотах (`csvPosition.quantity * lotSize`).

4. Сравнивает с балансом на сервере



Культура в компании

Что вы о ней думаете?

5. Количества не совпадают.
6. Позиция в CSV-файле отсутствует на сервере.

3. Безопасность робота:

- Любые обнаруженные расхождения вызывают ошибку, останавливающую торговые операции.
- Не позволяет роботу совершать сделки на основе устаревших или неверных данных.

Полный код `checkCSVpositions.js`:

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

const secrets = require('../../config/secrets'); // Ключи доступа и идентификаторы
const config = require('../../config/config'); // Параметры
const csvHandler = require('./csvHandler'); // Работа с CSV файлами

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

// Функция для получения всех позиций с сервера
async function getServerPositions() {
  try {
    const accountId = {
      accountId: secrets.AccountID
    };
    const response = await tinkoffClient.callApi('OperationsService/GetPositions',

    // Логруем полученные позиции с сервера
    logger.info(`Все открытые позиции счета ${secrets.AccountID}: \n ${JSON.stringify

    // Возвращаем только позиции с ценными бумагами (securities)
    return response.securities.map(sec => ({
      figi: sec.figi,
```



Культура в компании

Что вы о ней думаете?


```
        logger.error(`Ошибка при получении позиций с сервера: ${error.message}`);
        throw error;
    }
}

// Функция для проверки расхождений
async function checkForDiscrepancies() {
    try {
        // Загружаем текущие позиции из CSV файла
        var csvPositions = await csvHandler.loadPositions();

        // Получаем позиции с сервера
        const serverPositions = await getServerPositions();

        // Проверяем каждую позицию из CSV
        for (const csvPosition of csvPositions) {
            // Находим соответствующую позицию с сервера
            const serverPosition = serverPositions.find(pos => pos.figi === csvPosition.figi);

            if (serverPosition) {
                const lotSize = await tinkoffClient.getLot(csvPosition.figi);
                logger.info(`Количество бумаг в лоте ${csvPosition.figi}: ${lotSize} шт`);
                const csvTotal = csvPosition.quantity * lotSize;

                // Сравниваем количество позиций
                if (csvTotal !== serverPosition.balance) {
                    // Если есть расхождение, логируем ошибку и останавливаем торгового робота
                    logger.error(`Ошибка: Несоответствие по FIGI ${csvPosition.figi}. CSV баланс: ${csvTotal}, серверный баланс: ${serverPosition.balance}`);
                    throw new Error('Найдено несоответствие позиций. Остановка торговли');
                }
            } else {
                logger.error(`Ошибка: Позиция с FIGI ${csvPosition.figi} отсутствует на сервере`);
                throw new Error('Найдено несоответствие позиций. Остановка торговли.');
```



Культура в компании

Что вы о ней думаете?

```
// Экспортируем функции
module.exports = {
  checkForDiscrepancies
};

// checkForDiscrepancies().catch(logger.error);
```

Итоги

Проект полностью представлен на [Гитхабе](#). Новые модули будут загружаться по мере написания и тестирования.

Автор: Михаил Шардин

 [Моя онлайн-визитка](#)

 [Telegram «Умный Дом Инвестора»](#)

27 ноября 2024 г.

Теги: [московская биржа](#), [мосбиржа](#), [моех](#), [моexalgo](#), [tbank](#), [t-bank invest api](#)

Хабы: [Open source](#), [Финансы в IT](#), [JavaScript](#), [Node.JS](#)

Редакторский дайджест



Присылаем лучшие статьи раз в месяц

**178****70.7**

Карма

Рейтинг

Михаил Шардин [@empenoso](#)

[Автоматизация](#) / [Данные](#) / [Финансы](#) / [Умные дома](#)

[Подписаться](#)**Культура в компании**

Что вы о ней думаете?

 Комментарии 2

Публикации

ЛУЧШИЕ ЗА СУТКИ

ПОХОЖИЕ

**СОКРОВЕНЕУ**

14 часов назад

Первые впечатления от 1921bg015, отечественного RISC-V контроллера

**Культура в компании**

Что вы о ней думаете?

 +64 31 51

ru_vds

12 часов назад

Могут ли LLM писать более качественный код, если их об этом просто попросить?



Средний



19 мин



7.3K

Обзор

Перевод

 +40 46 8

SergioPrieto

17 часов назад

Алисе Селезневой — 60! Юбилей девочки, с которой ничего не случится



Простой



12 мин



9.2K

Обзор

 +38 18 56

13 часов назад

Зарплаты IT-специалистов во второй половине 2024: +19% к предыдущему полугодию



Простой



6 мин



17K

 +34 25 10

Maccimo

15 часов назад

Реверс-инжиниринг 128-битного дракона



12 мин



2.5K

Recovery Mode



Культура в компании

Что вы о ней думаете?

**PatientZero**

9 часов назад

Контрабанда данных внутри эмодзи

**Простой**

5 мин



1.9K

Обзор

Перевод

**+29**

31



12

**sergbe**

18 часов назад

Реальный кейс: 22 вопроса на техническом интервью на вакансию Java Lead в JP Morgan в 2025 году

**Сложный**

35 мин



9.2K

Тutorial

Перевод

**+27**

106



9

**rakill**

20 часов назад

Как я придумал геймификацию прокачки для мальчиков-геймеров в реальной жизни. Книга и настольная карточная игра

**Простой**

7 мин



2.3K

Тutorial

**+21**

23



7

**VGeydt**

11 часов назад

Карьера разработчика 2к25: берём вторую работу или помогаем заработать бизнесу?

**Простой**

7 мин



2.1K

Мнение

**Культура в компании**

Что вы о ней думаете?

**Jumse_Mary**

13 часов назад

Неубиваемый PHP: почему в 2025 году этот язык все еще остается одним из самых востребованных



Средний



3 мин



1.6K

Интервью



+19



14



32

Паглядзіце ўважліва: наш першы рейтинг ІТ-работодателей РБ

Турбо

[Показать еще](#)

ВАКАНСИИ

Бэкэнд-разработчик JavaScript

от 250 000 до 400 000 ₽ · Wanted. · Москва

JavaScript FullStack разработчик

от 80 000 до 130 000 ₽ · Rocket · Смоленск · Можно удаленно

Senior Frontend (JavaScript) разработчик

от 350 000 до 420 000 ₽ · Vital Partners · Можно удаленно

Разработчик JavaScript

от 130 000 до 180 000 ₽ · KeyHabbits · Москва · Можно удаленно

JavaScript FullStack developer

до 220 000 ₽ · Wanted. · Санкт-Петербург

[Больше вакансий на Хабр Карьере](#)

МИНУТОЧКУ ВНИМАНИЯ

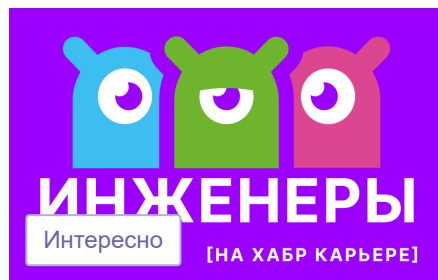


Культура в компании

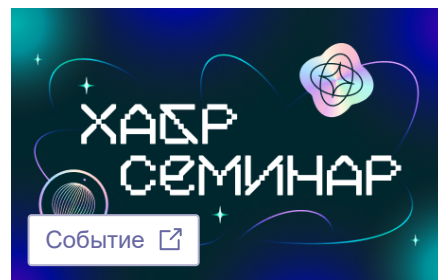
Что вы о ней думаете?



Техноархеолог отправляется в затерянный город и спасает мир



Инженеры тут? А если найдем? Новая специализация на ХК



PR продукта и IT-бренд — это мэтч: обсудим на Хабр Семинаре

РАБОТА

[Node.js разработчик](#)

48 вакансий

[JavaScript разработчик](#)

148 вакансий

[React разработчик](#)

50 вакансий

[Все вакансии](#)

БЛИЖАЙШИЕ СОБЫТИЯ



Культура в компании

Что вы о ней думаете?



14 – 19 февраля

Онлайн-интенсив «Нейросети для HR в 2025 году»

Онлайн

Другое

[Больше событий в календаре](#)

Хабр



Культура в компании

Что вы о ней думаете?

Настройка ссылки

Техническая поддержка

© 2006–2025, Habr



Культура в компании

Что вы о ней думаете?