



РЕКЛАМА

Хабр × K2 CLOUD

Разблокируй хабрареакты  
и участвуй в розыгрыше

empenoso

27 ноя в 03:23

## Отслеживание позиций торгового робота Московской биржи через CSV файл



Простой



7 мин



1.8K

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 Windows без использования виртуальной машины запустить несколько копий программы технического анализа с разными системами - для того, чтобы каждая из этих копий отправляла свои сигналы на покупку и продажу в соответствующий Квик. Это было нужно для разных торговых стратегий.

РЕКЛАМА



Реакции уже на Хабре

Разблокируй синих слизней

```

src > services > JS csvHandler.js > loadPositions > <function> > on('data') callback
15 const path = require('path'); // модуль для работы с путями, файлов и директорий
16 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
53   fs.writeFileSync(filePath, csvData);

```

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.
- **json2csv**, это утилита для преобразования данных JSON в формат CSV. Идеально подходит для экспорта данных из приложений в структуру, удобную для CSV, может работать как синхронно, так и асинхронно.

Установка этих библиотек:



Реакции уже на Хабре

Разблокируй синих слизней

## 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.
3. Возвращает обещание, которое разрешается с проанализированными данными или отклоняется в случае ошибки.

#### `savePositions(positions)`:

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



**Реакции уже на Хабре**

Разблокируй синих слизней

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 функции

// Загружаем все позиции из CSV файла
function loadPositions() {
  return new Promise((resolve, reject) => {
    const positions = [];
    fs.createReadStream(filePath)
      .pipe(csv())
      .on('data', (row) => {
        positions.push({
          ticker: row.ticker,
          figi: row.figi,
          quantity: parseFloat(row.quantity), // Преобразование количества в

```



**Реакции уже на Хабре**

Разблокируй синих слизней

```
        maxPrice: parseFloat(row.maxPrice), // Преобразование максимальной
        profitLoss: parseFloat(row.profitLoss) // Преобразование прибыли/убытка
    });
    });
    .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', 'profitLoss'];
    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
            positions.push(newPosition);
        } else {
            // Обновляем, если позиция уже существует
            positions[index] = newPosition;
        }

        savePositions(positions);
    });
}
```

**Реакции уже на Хабре**

Разблокируй синих слизней

```
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 основе неверных данных.

## Основные функции:

### 1. getServerPositions()



Реакции уже на Хабре

Разблокируй синих слизней

- 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.

Ошибки регистрируются, если:

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



**Реакции уже на Хабре**

Разблокируй синих слизней

- 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',

    // Логируем 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,
      balance: parseFloat(sec.balance) // Преобразуем баланс в float
    }));
  } catch (error) {
    logger.error(`Ошибка при получении позиций с сервера: ${error.message}`);
    throw error;
  }
}
```



Реакции уже на Хабре

Разблокируй синих слизней

```
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.');
    }
  }

  logger.info('Все позиции совпадают. Торговля продолжается.');
```

```
} catch (error) {
  logger.error(`Ошибка при проверке позиций: ${error.message}`);
  // Останавливаем торгового робота (добавьте здесь вашу логику остановки)
}

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

**Реакции уже на Хабре**

Разблокируй синих слизней

Итоги

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

Автор: [Михаил Шардин](#)

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

Электронпочта

→



168

Карма

56

Рейтинг

Михаил Шардин @empenoso

Разработчик

Подписаться



[Сайт](#) [Сайт](#) [Github](#) [Telegram](#)

РЕКЛАМА

Вы всё ближе к стабильной самооценке

промокод BLIZE активировать до 9.12

-20% на первые 3 сессии с психологом

18+ ЯСНО

Комментарии 2



Реакции уже на Хабре

Разблокируй синих слизней

ЛУЧШИЕ ЗА СУТКИ

ПОХОЖИЕ



DandyDan

18 часов назад

## Регулярные выражения простыми словами. Часть 2



Простой



10 мин



7.4K

Тutorial



+62



150



28



ptsecurity

18 часов назад

## DaMAgeCard: как индустрия карт памяти SD воскресила DMA-атаки



24 мин



3K

Кейс



+43



22



5



TShilova

10 часов назад

## Как вас обманывают в лазерной коррекции зрения: сравнение SMILE с CLEAR и SmartSight



Простой



12 мин



5.5K

Мнение



+35



76



50



Sugrob-57

17 часов назад

## Моки, L1 и никакого деплоя. Собираем идеальный .NET-тест для веб-сервиса



Средний



16 мин



1.1K



Реакции уже на Хабре

Разблокируй синих слизней

 +28 19 0**dsvdev**

14 часов назад

## Программисты ничего не знают (и это нормально)

**Простой**

5 мин



12K

Из песочницы

 +21 29 50**ru\_vds**

18 часов назад

## Как поднять свой сервер TeamSpeak

**Средний**

6 мин



2.2K

Тutorial

 +21 39 7**FirstJohn**

19 часов назад

## Домашние эксперименты с радиолампами. Часть 1. Теория



10 мин



2.7K

 +21 27 19**Squirrelfm**

12 часов назад

## Построение базы знаний компании и поиска документов на LLM и RAG

**Простой**

16 мин



1.9K

Tutorial

 +20 36 10**Реакции уже на Хабре**

Разблокируй синих слизней

**butivskiydm**

20 часов назад

## Про медвежий интеллект

**Простой**

6 мин



3.1K

Мнение

**+20**

10



39

**badcasedaily1**

22 часа назад

## Как избежать God Object в Java: несколько способов

**Простой**

5 мин



1.3K

Обзор

**+19**

13



0

## Рабочая схема: делимся рейтингом работодателей в IT за 2024 год

Турбо

Показать еще

### ИСТОРИИ

**Реакции уже на Хабре**

Разблокируй синих слизней



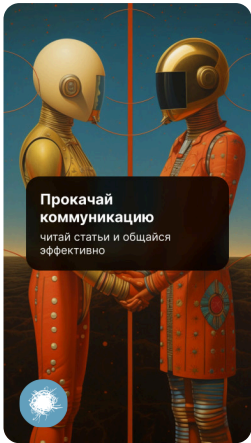
**Запустили спутник «Кондор-ФКА» № 2**



**Симфония технологий**



**Путешествие в мир ИИ**



**Прокачать коммуникацию**



**Перевернуть календарь и добавить событие**

## ЗАКАЗЫ

Доработка пользовательской части личного кабинета Vue 3

100000 руб./за проект · 12 откликов · 42 просмотра

Составить промт для qwen, позволяющий оперировать задачами из json

9999 руб./за проект · 5 откликов · 41 просмотр

Перевести файл из Фигмы в формат для печати

500 руб./за проект · 9 откликов · 71 просмотр

Фикс авторизации в плагине. REST API, React.js

2000 руб./за проект · 10 откликов · 64 просмотра

Подключить Python парсеры к PostgreSQL - TimescaleDB

30000 руб./за проект · 16 откликов · 81 просмотр

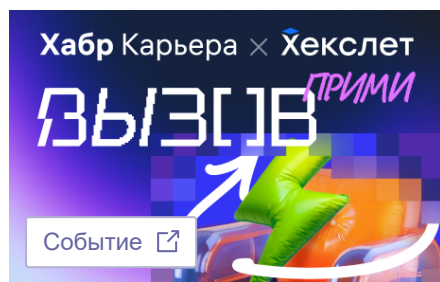
[Больше заказов на Хабр Фрилансе](#)

## МИНУТОЧКУ ВНИМАНИЯ



**Реакции уже на Хабре**

Разблокируй синих слизней



Бросаем вызов тем, кто хочет прокачать свои навыки



Исследуем новые миры: Хабр и ЭКОПСИ изучают IT-рынок РБ



Конференции, хакатоны, митапы: что посетить в декабре

## РАБОТА

[Node.js разработчик](#)

57 вакансий

[React разработчик](#)

43 вакансии

[JavaScript разработчик](#)

157 вакансий

[Все вакансии](#)

## БЛИЖАЙШИЕ СОБЫТИЯ



Реакции уже на Хабре

Разблокируй синих слизней



2 – 18 декабря

## Yandex DataLens Festival 2024

Москва • Онлайн

Разработка

Менеджмент

Аналитика

[Больше событий в календаре](#)

### Хабр



🌐 [Настройка языка](#)



**Реакции уже на Хабре**

Разблокируй синих слизней



© 2006–2024, Habr



### Реакции уже на Хабре

Разблокируй синих слизней