

Все важнейшие
IT-события здесь

5

6

7

8

9

10

11

12

13

14

15

16

17

18



empenoso

8 ноя 2024 в 08:33

Инструменты робота, торгующего на Московской бирже через API брокера



Сложный



8 мин



2K

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.

```
91 // } catch (error) {
92 //   logger.error('Ошибка ${stock}:', error.message);
93 // }
94 // }
95
96 // Получение понятного имени инструмента
97 for (const stock of config.securitiesToMonitorFigiArray) {
98   try {
99     const name = await tinkoffClient.getName(stock);
100     const nameUid = name.uid;
101     logger.info(`${name.nameCombination} это ${stock} или ${nameUid}.`);
102   } catch (error) {
103     logger.error('Ошибка ${stock}:', error.message);
104   }
105 }
106
107 // // Тест корректности размера лотов:
108 // const figi = 'BBG0047308P8'; // Пример ФИГИ
109 // const price = await tinkoffClient.getQuote(figi);
```

```
[Running] node "d:\Synology\SilverFir-TradingBot\src\instruments.js"
2024-11-08 10:11:27 [INFO]: Запуск функции "test"
2024-11-08 10:11:27 [INFO]: Запуск функции "instruments"
2024-11-08 10:11:27 [INFO]: Банк ВТБ (VTBR) это BBG004730ZJ9 или 8e2b0325-0292-4654-8a18-4f63ed3b0e09.
2024-11-08 10:11:27 [INFO]: Мечел (MTLR) это BBG004568598 или eb4ba863-e85f-4f80-8c29-f2627938ee58.
2024-11-08 10:11:27 [INFO]: ОБК (UMGN) это BBG0008D3V85 или a8dc188b-cae8-40c0-99e8-f561f4339751.
2024-11-08 10:11:28 [INFO]: РуссНефть (RNFT) это BBG00F9XX7H4 или c7485564-ed92-45fd-a724-1214aa202904.
2024-11-08 10:11:28 [INFO]: ЕвроТранс (EUTR) это TCS00A1002V2 или 02b2ea14-3c4b-47e8-9548-45a8dbcc8f8a.
2024-11-08 10:11:28 [INFO]: Сургутнефтегаз - привилегированные акции (SNGSP) это BBG0045681M2 или a797f14a-8513-4b84-b15e-a3b98dc4cc00.
2024-11-08 10:11:28 [INFO]: Газпром (GAZP) это BBG0047308P8 или 962e2a95-02a9-4171-abd7-aa198db0643a.
2024-11-08 10:11:28 [INFO]: Роснефть (ROSN) это BBG004731354 или fd417230-19cf-4e7b-9623-f7c9ca18ec6b.
2024-11-08 10:11:28 [INFO]: Сбер Банк (SBER) это BBG004730N88 или e6123145-9665-43e0-8413-cd61b8aa9b13.
2024-11-08 10:11:29 [INFO]: Сегежа (SGZH) это BBG0100R9963 или 7bedd86b-478d-4742-a28c-29d27f8dbc7d.
2024-11-08 10:11:29 [INFO]: Аэрофлот (AFLT) это BBG0045683M7 или 1c69e020-f3b1-455c-affa-45f8b0049234.
2024-11-08 10:11:29 [INFO]: ВК (VKCO) это TCS00A100VFE или b71bd174-c72c-41b0-a66f-5f9073e0d1f5.
2024-11-08 10:11:29 [INFO]: РУСАЛ (RUAL) это BBG008F21312 или f866872b-0f68-4b6e-930f-749f9e9a79c0.
2024-11-08 10:11:29 [INFO]: Татнефть (TATN) это BBG004RVFFC0 или 88468f6c-c67a-4fb4-a006-53eed03883c.
[Done] exited with code=0 in 2.793 seconds
```

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 данным 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 доходности

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

const TinkoffClient = require('../grpc/tinkoffClient'); // модуль для взаимодействия с API
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/GetInstrumentBy
  // 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

 [Моя онлайн-визитка](#)

 [Telegram «Умный Дом Инвестора»](#)

11 ноября 2024 г.

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

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

Редакторский дайджест



Присылаем лучшие статьи раз в месяц

**183**

Карма

87.1

Рейтинг

Михаил Шардин @empenoso

Автоматизация / Данные / Финансы / Умные дома

[Подписаться](#)

[Сайт](#) [Сайт](#) [Github](#) [Комментировать](#)

Публикации

[ЛУЧШИЕ ЗА СУТКИ](#)[ПОХОЖИЕ](#)**rssdev10**

22 часа назад

Почему въехав по «визе талантов» в США я с радостью вернулся в Россию

 Средний  32 мин  34K

Мнение

 +173

 109

 494



melnik909

18 часов назад

Вы не знаете CSS. Мои вопросы о CSS с ответами. Часть 2

 Средний  7 мин  2K

Обзор

 +38

 31

 1



DAN_SEA

16 часов назад

Генерация случайных чисел

 Средний  10 мин  2.3K

Обзор

 +31

 23

 33



OrkBiotecnologist

22 часа назад

VPS за 139 рублей — дом для вашего резюме на основе Hugo

 Простой  7 мин  8.2K

Тutorial

 +28

 45

 12



nebykoff_anton

23 часа назад

До и после: оптимизация изображений для Lighthouse и не только

 Простой  8 мин  946

Тutorial

 +28 24 2**Корcheniy**

23 часа назад

Имитатор касаний. Ч2: Железная часть



Средний



9 мин



1.7K

Кейс

 +24 20 12**PatientZero**

2 часа назад

Пишем стек TCP/IP с нуля: Ethernet, ARP, IPv4 и ICMPv4



Простой



13 мин



791

Тutorial

Перевод

 +18 24 1**tertiumnon**

17 часов назад

Минимум книг, которые нужно прочитать начинающему или продолжающему свою кривую обучения программисту



Простой



3 мин



8.4K

Обзор

 +17 184 19**lbkanter**

55 минут назад

Бэкдор Auto-color: разбор угрозы, технический анализ и способы защиты



Средний



4 мин



236

Обзор

 +13 5 2



alexander-shustanov

18 часов назад

В поисках идеального Database-клиента для IDE: Amplicode выбирает DBeaver



Простой



6 мин



2.5K



+13



13



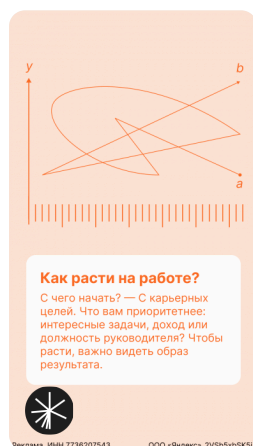
7

Бегущий по лезвию ИИ — 2025: сезон футурологии на Хабре с крутыми призами

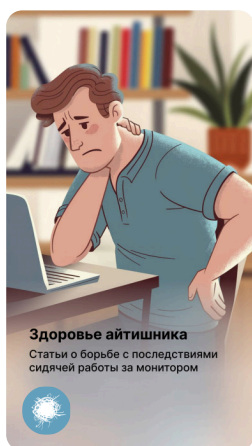
Турбо

Показать еще

ИСТОРИИ



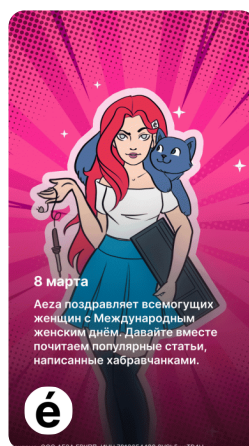
Как расти на работе?



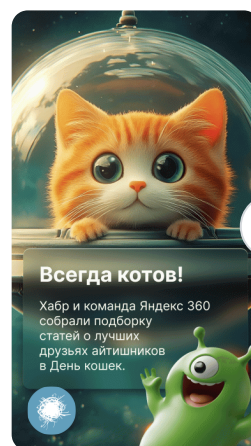
Здоровье айтишника



Угадайте будущее в новом сезоне



С праздником весны!

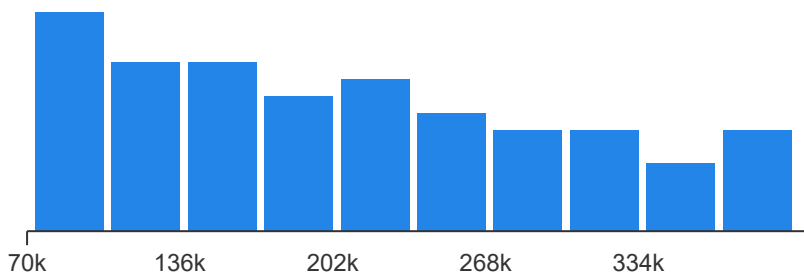


Всегда котов!

СРЕДНЯЯ ЗАРПЛАТА В IT

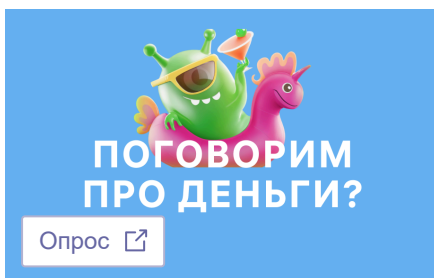
212 662 ₽/мес.

— средняя зарплата во всех IT-специализациях по данным из 33 637 анкет, за 1-ое пол. 2025 года. Проверьте «в рынке» ли ваша зарплата или нет!

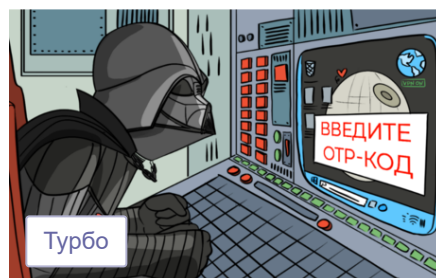


[Проверить свою зарплату](#)

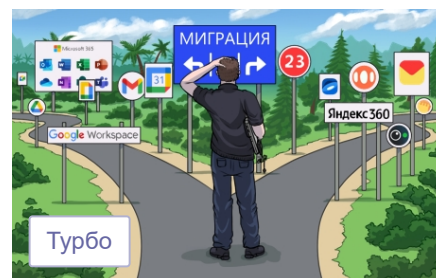
МИНУТОЧКУ ВНИМАНИЯ



Как айтишники подходят к финансовому планированию?



«Люк, я твой фактор!» — защищаем подключения с MFA



Как избежать ошибок при бизнес-миграции и сохранить данные?

РАБОТА

[React разработчик](#)

48 вакансий

[JavaScript разработчик](#)

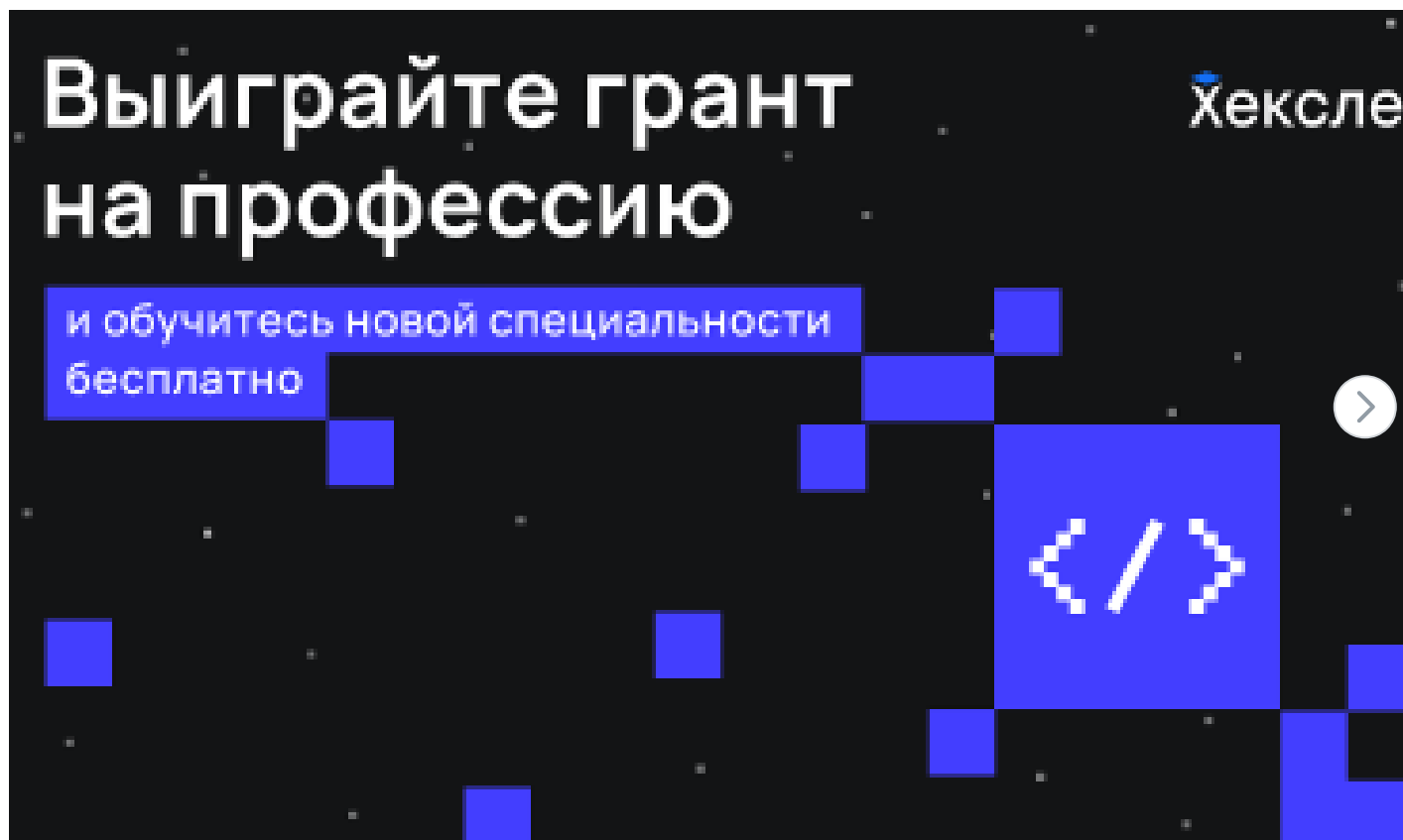
134 вакансии

[Node.js разработчик](#)

56 вакансий

[Все вакансии](#)

БЛИЖАЙШИЕ СОБЫТИЯ



17 февраля – 24 марта

Конкурс «Снежный код» от Хекслета. Три гранта на бесплатное 10-месячное обучение

Онлайн

Разработка

[Больше событий в календаре](#)

Хабр



🌐 [Настройка языка](#)

[Техническая поддержка](#)

© 2006–2025, Habr