Руководство для начинающих по инструментам разработки Node.JS для создания приложений нового поколения

10.04.2023

Десятилетие назад, когда Райан Дал написал Node.js, чтобы создать решение, решающее проблемы скорости, параллелизма, масштабируемости, надежности и обмена данными, которые часто встают перед разработчиками, он совершенно не подозревал о популярности и огромном успехе этого программного продукта на более позднем этапе. Даже опрос разработчиков stack overflow признал этот язык наиболее часто используемой технологией back-end в мире ИТ.

Сейчас платформа с открытым исходным кодом, построенная на базе JavaScript-движка V8 Chrome и обеспечивающая работу почти 0,49% веб-сайтов в мире с 110 945 уникальными доменами, уверенно продвигается вверх по уровню использования и не показывает никаких признаков того, что проигрывает битву современным фреймворкам.

За последние несколько лет экосистема пакетов Node.js, а именно NPM, может похвастаться самой большой экосистемой библиотек с открытым исходным кодом в мире с миллионами пакетов. Растущая поддержка сообщества, универсальная экосистема разработки и высокая степень знакомства с языком JavaScript – другие причины использования Node.JS для веб-разработки.

Непревзойденная производительность, высокая масштабируемость, разработка приложений в реальном времени, единая кодовая база, сервисы потоковой передачи данных и неблокируемая модель ввода-вывода, ориентированная на события, заставили как

крупные предприятия, так и малый и средний бизнес создавать или реорганизовывать веб-сайты и приложения с использованием технологии Node.js. Netflix, Uber, PayPal и Walmart – вот лишь несколько названий.

Это достаточные причины для разработчиков изучить или отточить свои навыки в Node.js, чтобы получить работу своей мечты в лучших компаниях по разработке Node.js. Новый кандидат в технологическом блоке облегчил жизнь разработчикам Node.js благодаря множеству инструментов разработки Node.js. Больше – значит лучше.

Что такое Node.JS?

Node.JS – это легкая серверная платформа JavaScript, построенная на базе движка Google Chrome V8 JS. JavaScript и TypeScript – оба языка кодирования используются для создания приложений Node.JS. Веб-приложение Node.JS использует модель неблокирующего ввода-вывода, управляемую событиями, что делает его подходящим для создания высокопроизводительных приложений реального времени. Именно поэтому платформу используют такие ведущие бренды, как Twitter, eBay, PayPal, Netflix, aliexpress и другие.

Низкая кривая обучения и огромная поддержка сообщества позволяют начинающим программистам быстро освоить и начать создавать современные приложения с помощью Node.JS. Высокая расширяемость и масштабируемость позволяют талантливым разработчикам удовлетворять индивидуальные требования бизнес-проектов. Именно поэтому веб-разработка на Node.JS становится приоритетным выбором для бизнеса.

Что такое инструменты разработки Node.JS?

С 2009 года Node.JS завоевал высокую популярность благодаря постоянным обновлениям, и сейчас этот JS-фреймворк находится в

хорошем положении, так что от стартапов до предприятий предпочитают его другим JS-фреймворкам. Однако разработка Node.JS не является простым и легким путем для разработчиков. Именно здесь на помощь приходят инструменты для разработки Node JS.

Инструменты для разработчика Node.JS – это полный пакет, включающий библиотеки, фреймворки и инструменты, которые облегчают разработку, проектирование, тестирование и развертывание. Разработчики Node.js используют инструменты в зависимости от потребностей проекта.

Преимущества использования инструментов разработки Node.JS

В каждой технологии есть свои плюсы и минусы. И Node.JS не является исключением. Здесь мы рассмотрим преимущества Node.JS, чтобы знать, когда и где он соответствует потребностям вашего бизнеса и позволит предприятиям сделать правильный выбор.

- Однопоточная, событийно-ориентированная архитектура позволяет приложению обрабатывать множество одновременных запросов со скоростью, не засоряя оперативную память, что приводит к созданию приложений реального времени с высокой производительностью.
- Кластерный модуль, балансировка нагрузки, неблокирующий механизм event-loop и использование микросервисов позволяют создавать масштабируемые приложения.
- Разработчики Node.JS могут писать код для front-end и back-end, используя один и тот же язык JavaScript, что устраняет необходимость найма дополнительных ресурсов для разработки веб-сайтов и приложений. Это делает разработку полностекевого JS экономически эффективной.
- Активный вклад большого количества опытных экспертов по разработке Node.JS помогает начинающим разработчикам

получить помощь, когда это необходимо.

- JavaScript (JS) является универсальным языком программирования, который легче изучить, что облегчает разработчикам Node.JS написание кода в сокращенные сроки.
- Возможности полностекевого Node. JS позволяет межфункциональной команде сосредоточиться на жизненном цикле разработки и решать проблемы немедленно.
- Легкий Node.JS ускоряет скорость разработки благодаря одинаковому использованию JS для разработки на стороне клиента и на стороне сервера.
- Встроенные API-средства для разработки Node.JS позволяют создавать HTTP- и DNS-серверы и JSON, которые упрощают обмен данными между веб и клиент-сервером. Это позволяет удовлетворять индивидуальные требования бизнес-проектов.
- Модуль кэширования минимизирует нагрузку на задачи и повторное выполнение кода, что приводит к загрузке страницы за доли секунды.

Лучшие инструменты для разработки Node.JS

Bluebird.js

Это полнофункциональная библиотека JS, которая позволяет создавать и предоставлять непревзойденные по производительности приложения. Инструмент делает написание приложений Node.JS доступным, поскольку асинхронное использование модулей node становится возможным. Существует концепция ‘Promisify’, которая используется для функции обратного вызова, чтобы убедиться, что каждый вызов функции обратного вызова возвращает некоторое значение. Таким образом, все функции под зонтиком узлового модуля автоматически изменяются и возвращают некоторое значение, когда узловые модули ‘Promisify’.

Electrode.JS

Electrode.JS – это универсальная платформа, на которой выполняются программы Node.JS. Будучи стандартизированным фреймворком, использующим новейшие технологии, Electrode.JS позволяет разработчикам Node.js просматривать дерево модулей, документацию и многое другое. Он больше ориентирован на создание высокопроизводительных приложений с возможностью повторного использования компонентов и легким развертыванием. Он позволяет разработчикам создавать уникальные приложения Node.JS.

Mocha.js

Опытные разработчики Node.JS используют фреймворк тестирования в масштабе, чтобы сделать асинхронное тестирование простым. Фреймворк тестирования JS работает одновременно на Node.JS и в браузере. Он позволяет запускать несколько тестов Node.JS параллельно, создавать отчеты о покрытии тестов, обнаруживать утечки глобальных переменных, поддерживать конфигурационные файлы и многое другое. Инструмент тестирования используется для модульного и интеграционного тестирования приложений Node.JS.

Webstorm IDE

IDE для JS-фреймворка делает разработку легкой благодаря автоматизации рутинной работы и обработке сложных задач. Инструмент Node.JS позволяет писать надежный и сопровождаемый код, поскольку он мгновенно обнаруживает потенциальные проблемы. Кроме того, вся кодовая база рефакторингуется в несколько кликов, не пропуская ничего при огромных структурных изменениях. Отладка и тестирование Node.JS с помощью WebStorm IDE – это простое и легкое путешествие.

Express.js

Express – это серверный фреймворк, написанный на языке JavaScript и считающийся идеальным решением для создания одной страницы, нескольких страниц и гибридных приложений, общих внутренних функций и API. Он поддерживает архитектуру MVC и имеет два шаблонизатора, которые обрабатывают поток данных. В дальнейшем он предлагает механизм добавления дополнительного “промежуточного ПО” для обработки запросов в цепочку обработки запросов в любой точке, что обеспечивает гибкость разработчикам для создания совместимых пакетов промежуточного ПО, которые решают проблемы разработки.

Однако гибкость в добавлении пакетов промежуточного ПО в любой момент работает как меч о двух концах, потому что нет правильного способа использовать пакеты промежуточного ПО и структурировать приложение в один файл, несколько файлов или любую структуру каталогов. Именно поэтому фреймворк также называют недизайнерским и минималистским. Используя Express-генератор, можно сгенерировать структуру папок проекта, чтобы разработчики Nodejs могли сразу приступить к кодированию.

Тонкий слой поверх Node.JS с функциями веб-приложения, такими как промежуточное ПО, маршрутизация, обслуживание статических файлов и шаблонизация, обеспечивает высокую производительность ввода-вывода приложения Node.JS. Кроме того, Express предоставляет методы для определения того, какая функция вызывается для определенного HTTP-глагола и шаблона URL, какой шаблонизатор используется, где находятся файлы шаблонов и какой шаблон использовать для получения ответа. Пакеты промежуточного ПО и методы HTTP-утилиты позволяют создавать API на лету.

Socket.io

Когда необходимо создать высокоинтерактивное приложение для чата в реальном времени, фреймворк socket.io оказывает большую помощь, обеспечивая двунаправленный синхронный обмен данными в

реальном времени. Простые события с одним аргументом позволяют легко интегрироваться с любой технологией реактивной потоковой передачи данных и использовать отслеживание событий на основе причинно-следственной последовательности.

Кроме того, socket.io совместим со всеми браузерами, не зависит от платформы и безупречно работает и функционирует на любом устройстве. В движке реального времени также реализованы функции аналитики в реальном времени (отображение данных в виде графиков, журналов или счетчиков реального времени), мгновенного обмена сообщениями и чата (создание простого приложения за несколько строк кода), бинарной потоковой передачи (возможность обмена текстом, изображениями, аудио или видео) и совместной работы с документами (возможность одновременного внесения и просмотра изменений несколькими пользователями), что сделало его очень популярным и стоит того, что его используют Trello, Zendesk, Microsoft Office и многие другие популярные организации.

Meteor

Этот полнофункциональный фреймворк является отличным выбором для разработки реактивных веб- и мобильных приложений. Поддержка принципа проектирования MVC облегчает разработчикам создание приложения, в котором слои пользовательского интерфейса и бизнес-логики не различаются, и тем самым обеспечивает обновление данных в реальном времени без дополнительных обращений к веб-серверу. Благодаря механизму двунаправленной синхронизации данных сохраняется теневая копия данных, запрашиваемых клиентом с сервера, а при внесении изменений они автоматически сохраняются на сервере без какого-либо ручного вмешательства.

Высокопроизводительные и сложные приложения могут быть созданы быстро благодаря обилию вариантов строительных лесов, библиотек и пакетов. С инструментами тестирования velocity тестирование кода начинается сразу после его написания, а зеленые или красные точки показывают результаты тестирования.

Фреймворк, поддерживаемый активным сообществом, постоянно развивается, пополняясь новыми библиотеками, пакетами и инструментами, чтобы соответствовать новым тенденциям в разработке приложений.

Chai

Node.JS привнес язык JavaScript в программирование на стороне сервера, программирование на стороне клиента или программирование встроенных систем. Программное обеспечение, написанное на JavaScript, должно быть протестировано с помощью тестовых фреймворков, включающих бегунки тестирования, утверждения или наборы тестов в стиле BDD.

Инструмент Chai выделяется как библиотека утверждений BDD/TDD, которая может быть сопряжена с любым фреймворком тестирования JavaScript, будь то Mocha, Sinon или Jasmine. Спецификация языка BDD/TDD для всех утверждений описана в документации API chai, и разработчики могут выбрать любое из утверждений, с которым им удобнее работать. Кроме того, утверждения chai могут быть расширены на новые контексты с помощью плагинов, и разработчики могут создавать свои собственные плагины или использовать существующий шаблон плагина.

Keystone.js

Использование CMS для создания сложных веб-приложений экономит много времени и усилий на разработку, поскольку готовые плагины не требуют от разработчиков Node.js кодирования с нуля для создания и развертывания функциональных возможностей. Keystone – это CMS, которая помогает в создании приложений, управляемых данными, которые взаимодействуют с базой данных без учета платформы PHP или WordPress. По сути, keystone.js основана на Express.js и MongoDB, где разработчикам требуется только настроить архитектуру MVC после установки. Модели данных в CMS автоматически создают приборную панель администратора для управления данными. Полный контроль над системой изнутри позволяет настраивать ее “на лету”. Легкая

интеграция пакетов делает CMS расширяемой.

> *“Когда вы нанимаете разработчиков, готовые шаблоны и высокая степень настройки делают создание блогов или электронной коммерции делом нескольких часов”.*

Инструмент поставляется с функциями управления сеансами и аутентификации, которые снимают тяжелую работу с плеч разработчика. Совместимость со сторонними сервисами является дополнением к готовым приложениям. Это делает keystone оптимальным решением для создания полноценных, высокопроизводительных и динамичных приложений в условиях сжатых сроков.

Koa.js

Команда, стоящая за фреймворком Express, создала koa, чтобы минимизировать минимализм, характерный для Express, не путем объединения любого промежуточного ПО в ядро Koa, а путем создания отдельных модулей для различных функций, которые могут быть добавлены по мере необходимости. Именно поэтому основной модуль Koa состоит всего лишь из 2K строк кода, что позволяет сократить объем загружаемого кода.

С использованием контекстного объекта ctx, опыт разработчика в написании промежуточных процедур перевернулся с ног на голову. Код, написанный с использованием async/await в koa, короткий, выразительный, легко читаемый и простой в сопровождении. Koa имеет небольшое преимущество перед фреймворком Express.js, когда речь идет о производительности. Koa можно расширить функциональными возможностями с помощью плагинов и промежуточного ПО. Интеграция генератора делает его хорошим в обработке ошибок. Кроме того, необходимость в написании дополнительного кода для потоковой передачи и закрытия файла отпадает, поскольку koa напрямую передает файл вместо потоковой передачи.

Sail.js

Популярный MVC-фреймворк, эмулирующий паттерн MVC фреймворка Ruby On Rails, обеспечивает поддержку нескольких баз данных (в комплекте с мощным ORM и Waterline), а также WebSockets для создания приложений для чата с интенсивным использованием данных в режиме реального времени. Он автоматически генерирует код для моделей, контроллеров и маршрутов, которые могут быть настроены позже, чтобы наилучшим образом соответствовать специфическим потребностям бизнеса.

Совместимость с CSRF и Lusca делает приложение надежным и безопасным с различными уровнями защиты. Отличная совместимость с множеством front-end фреймворков, таких как Angular, React, iOS, Android или Windows Phone, делает sail агностиком front-end. Промежуточная степень автоматизации, такая как автогенерация REST API, позволяет разработчикам сосредоточиться на логике на стороне сервера и запустить бэкенд без написания лишних строк кода, что ускоряет разработку и обеспечивает масштабируемость без сбоев в производительности.

PM2

Для управления приложениями Node.JS на рабочем сервере необходим инструмент, который решает технические проблемы и эффективно развертывает приложение. Именно здесь пригодится инструмент PM2. PM2 – это менеджер процессов Node.js с открытым исходным кодом, который помогает разработчикам управлять приложениями в производственной среде наилучшим образом посредством обработки процессов и ошибок узла, автоматической балансировки нагрузки, системы развертывания и мониторинга, а также декларативной конфигурации приложений.

Кроме того, PM2 отслеживает необработанные исключения, такие как неожиданные ошибки, исключения и сбои, и когда они возникают, он перезагружает процесс узла и отправляет электронное письмо или предупреждение, когда процесс

завершается или перезапускается. Режим кластера повышает производительность приложения за счет запуска экземпляров и автоматической балансировки нагрузки между каждым экземпляром. Он облегчает автоматизированное развертывание, когда разработчикам требуется только настроить приложение в файле, а обо всем остальном позаботится PM2. Встроенный балансировщик нагрузки делает масштабирование приложения простым делом. Используя PM2, можно вносить обновления в приложение в режиме реального времени, перезагружая процессы с нулевым временем простоя. Таким образом, PM2 снимает с вас огромные накладные расходы по поддержанию работоспособности приложения.

Babel

При разработке фронтенда разработчикам дорого писать код для ES6 или ES7, а также для всех старых версий, которые работают в браузерах и на устройствах. Тем не менее, большинство устройств и браузеров поддерживают версии ES5 и старше, что делает невозможным игнорирование совместимости старых версий. Это создает необходимость в транспойлере, который позволяет разработчикам писать код в соответствии с соглашениями последней версии, а затем автоматически транспонировать код JavaScript на обратно совместимые версии JavaScript.

Существует множество транспиляторов, но наиболее предпочтительным является транспилятор babel, поскольку он наиболее близок к ванильному JavaScript, уменьшает размер кода, имеет высокую совместимость с большинством браузеров, поддерживает карту для легкой отладки кода и, наконец, устраняет беспокойство об ошибках из-за несовместимости. В последнем обновлении, babel 7, транспилятор оснащен функциями повышения производительности, переопределениями конфигурации, конфигурациями JavaScript, опциями минификации размера, автоматического заполнения поли, метаданными вызывающего пользователя, поддержкой TypeScript и фрагментов React JSX, что делает его достойным использования при разработке приложений Node.js.

Broccoli

Когда приложение разрабатывается с использованием Grunt, и любой файл изменяется в сборке несколько месяцев спустя, приложение загружается более десяти секунд после обновления браузера. Двузначная задержка нетерпима и влияет на производительность разработчика. Чтобы сократить время пересборки, был внедрен плагин broccoli- grunt, который обеспечивает время пересборки менее 0,5 секунды и упрощает сценарии сборки. Разработчикам Node js нужно просто запустить задачи в Grunt, а затем broccoli создает сборку быстрым и простым способом.

По сути, broccoli – это дерево, которое не передает каталоги файловой системы, что позволяет компилятору n:1 легко справляться с каламбуром дерева. Отказ от создания частичной сборки, создает свежую сборку каждый раз и позволяет плагинам кэшировать вывод, так что большая часть сборки возвращается плагином из кэша. Инструмент сборки не зависит от бэкенда и обеспечивает поддержку пересборки в постоянном времени и компактных определений сборки. Быстрый и надежный конвейер активов, основанный на модуле ES6, прекрасно работает при разработке приложений.

Webpack

В настоящее время для разработки приложений пишутся десятки скриптов, каждый из которых имеет свою цель и требования, а также имеет зависимости. Разработчики должны загружать скрипты в правильном порядке, но асинхронная природа JavaScript по умолчанию пытается загрузить каждый скрипт сразу, что многократно увеличивает проблему. Именно поэтому сообщество создало модули для объединения скриптов в пакет, но медленно развивающийся JavaScript затрудняет продвижение. Здесь на помощь приходит загрузчик модулей, такой как Webpack.

Webpack – это инструмент сборки, который проходит через JavaScript для определения приоритета загрузки, зависимостей,

путей, эффективности и так далее без каких-либо накладных расходов. Он невероятно полезен в больших приложениях со сложным front-end благодаря меньшему количеству статических активов, легкому разделению кода, устранению мертвых активов, стабильному деплою на производстве и полному контролю над обработкой активов.

Выберите идеальные инструменты разработки Node.js

Важность технологических стеков для разработки программного обеспечения нельзя недооценивать, поскольку это может сделать или сломать успех проекта. Есть одна технология – Node.js, которая берет штурмом пространство разработки веб-сайтов, приложений и программного обеспечения. Инвестиции в кодирование Node.js – это хороший выбор сейчас и в ближайшие годы, поскольку технология пользуется спросом и занимает довольно высокие позиции. Когда вы будете готовы воспользоваться ошеломляющим успехом платформы разработки Node.JS, вам на помощь придет огромное количество инструментов разработки. Теперь, когда вы получили базовое представление об инструментах, вы можете подумать о найме разработчиков Node.js из списка компаний, предлагающих лучшие в своем классе услуги, для разработки надежного бизнес-решения.

Часто задаваемые вопросы

Чем хорош Node.Js?

Node.JS хорош тем, что позволяет создавать приложения, интенсивно использующие данные и работающие в режиме реального времени, которые могут обрабатывать большое количество соединений, работая на распределенных устройствах.

Какие средства разработки устанавливаются вместе с Node.Js?

Meteor.JS, Express.JS, Keystone.JS, PM2, Babel и другие инструменты разработки устанавливаются вместе с Node.JS, что облегчает разработку.