Мы начинаем публикацию серии материалов, которые представляют собой поэтапный перевод руководства по Node.js для начинающих. А именно, в данном случае «начинающий» — это тот, кто обладает некоторыми познаниями в области браузерного JavaScript. Он слышал о том, что существует серверная платформа, программы для которой тоже пишут на JS, и хотел бы эту платформу освоить. Возможно, вы найдёте здесь что-то полезное для себя и в том случае, если уже знакомы с Node.js.

Кстати, в прошлом году у нас был похожий по масштабам проект, посвящённый bash-скриптам. Тогда мы, после публикации всех запланированных материалов, собрали их в виде PDF-файла. Так же планируется поступить и в этот раз.

Сегодня мы обсудим особенности Node.js, начнём знакомство с экосистемой этой платформы и напишем серверный «Hello World».

Обзор Node.js

Node.js — это опенсорсная кроссплатформенная среда выполнения для JavaScript, которая работает на серверах. С момента выпуска этой платформы в 2009 году она стала чрезвычайно популярной и в наши дни играет весьма важную роль в области веб-разработки. Если считать показателем популярности число звёзд, которые собрал некий проект на GitHub, то Node.js, у которого более 50000 звёзд, это очень и очень популярный проект.

Платформа Node.js построена на базе JavaScript движка V8 от Google, который используется в браузере Google Chrome. Данная платформа, в основном, используется для создания веб-серверов, однако сфера её применения этим не ограничивается.

Рассмотрим основные особенности Node.js.

Скорость

Одной из основных привлекательных особенностей Node.js является скорость. JavaScript-код, выполняемый в среде Node.js, может быть в два раза быстрее, чем код, написанный на компилируемых языках, вроде C или Java, и на порядки быстрее интерпретируемых языков наподобие Python или Ruby. Причиной подобного является неблокирующая архитектура платформы, а конкретные результаты зависят от используемых тестов производительности, но, в целом, Node.js — это очень быстрая платформа.

Простота

Платформа Node.js проста в освоении и использовании. На самом деле, она прямо-таки очень проста, особенно это заметно в сравнении с некоторыми другими серверными платформами.

JavaScript

В среде Node.js выполняется код, написанный на JavaScript. Это означает, что миллионы фронтенд-разработчиков, которые уже пользуются JavaScript в браузере, могут писать и серверный, и клиентский код на одном и том же языке программирования без необходимости изучать совершенно новый инструмент для перехода к серверной разработке.

В браузере и на сервере используются одинаковые концепции языка. Кроме того, в Node.js можно оперативно переходить на использование новых стандартов ECMAScript по мере их реализации на платформе. Для этого не нужно ждать до тех пор, пока пользователи обновят браузеры, так как Node.js — это серверная среда, которую полностью контролирует разработчик. В результате новые возможности языка оказываются доступными при установке поддерживающей их версии Node.js.

Движок V8

В основе Node.js, помимо других решений, лежит опенсорсный JavaScript-движок V8 от Google, применяемый в браузере Google Chrome и в других браузерах. Это означает, что Node.js пользуется наработками тысяч инженеров, которые сделали среду выполнения JavaScript Chrome невероятно быстрой и продолжают работать в направлении совершенствования V8.

Асинхронность

В традиционных языках программирования (C, Java, Python, PHP) все инструкции, по умолчанию, являются блокирующими, если только разработчик явным образом не позаботится об асинхронном выполнении кода. В результате если, например, в такой среде, произвести сетевой запрос для загрузки некоего JSON-кода, выполнение потока, из которого сделан запрос, будет приостановлено до тех пор, пока не завершится получение и обработка ответа.

JavaScript значительно упрощает написание асинхронного и неблокирующего кода с использованием единственного потока, функций обратного вызова (коллбэков) и подхода к разработке, основанной на событиях. Каждый раз, когда нам нужно выполнить тяжёлую операцию, мы передаём соответствующему механизму коллбэк, который будет вызван сразу после завершения этой операции. В результате, для того чтобы программа продолжила работу, ждать результатов выполнения подобных операций не нужно.

Подобный механизм возник в браузерах. Мы не можем позволить себе ждать, скажем, окончания выполнения AJAX-запроса, не имея при этом возможности реагировать на действия пользователя, например, на щелчки по кнопкам. Для того чтобы пользователям было удобно работать с веб-страницами, всё, и загрузка данных из сети, и обработка нажатия на кнопки, должно происходить одновременно, в режиме реального времени.

Если вы создавали когда-нибудь обработчик события нажатия на кнопку, то вы

уже пользовались методиками асинхронного программирования.

Асинхронные механизмы позволяют единственному Node.js-серверу одновременно обрабатывать тысячи подключений, не нагружая при этом программиста задачами по управлению потоками и по организации параллельного выполнения кода. Подобные вещи часто являются источниками ошибок.

Node.js предоставляет разработчику неблокирующие базовые механизмы ввода вывода, и, в целом, библиотеки, использующиеся в среде Node.js, написаны с использованием неблокирующих парадигм. Это делает блокирующее поведение кода скорее исключением, чем нормой.

Когда Node.js нужно выполнить операцию ввода-вывода, вроде загрузки данных из сети, доступа к базе данных или к файловой системе, вместо того, чтобы заблокировать ожиданием результатов такой операции главный поток, Node.js инициирует её выполнение и продолжает заниматься другими делами до тех пор, пока результаты выполнения этой операции не будут получены.

Библиотеки

Благодаря простоте и удобству работы с менеджером пакетов для Node.js, который называется npm, экосистема Node.js прямо-таки процветает. Сейчас в реестре npm имеется более полумиллиона опенсорсных пакетов, которые может свободно использовать любой Node.js-разработчик.
Рассмотрев некоторые основные особенности платформы Node.js, опробуем её в действии. Начнём с установки.

Установка Node.js

Node.js можно устанавливать различными способами, которые мы сейчас рассмотрим.
Так, официальные установочные пакеты для всех основных платформ можно найти здесь.

Существует ещё один весьма удобный способ установки Node.js, который заключается в использовании менеджера пакетов, имеющегося в операционной системе. Например, менеджер пакетов macOS, который является фактическим стандартом в этой области, называется Homebrew. Если он в вашей системе есть, вы можете установить Node.js, выполнив эту команду в командной строке:

```
brew install node
```

Список менеджеров пакетов для других операционных систем, в том числе — для Linux и Windows, можно найти здесь.

Популярным менеджером версий Node.js является nvm. Это средство позволяет удобно переключаться между различными версиями Node.js, с его помощью можно, например, установить и попробовать новую версию Node.js, после чего, при необходимости, вернуться на старую. Nvm пригодится и в ситуации, когда нужно испытать какой-нибудь код на старой версии Node.js.

Я посоветовал бы начинающим пользоваться официальными установщиками Node.js. Пользователям macOS я порекомендовал бы устанавливать Node.js с помощью Homebrew. Теперь, после того, как вы установили Node.js, пришло время написать «Hello World».

Первое Node.js-приложение

Самым распространённым примером первого приложения для Node.js можно назвать простой веб-сервер. Вот его код:

```
const http = require('http')
const hostname = '127.0.0.1'
const port = 3000
const server = http.createServer((req, res) => {
  res.statusCode = 200
  res.setHeader('Content-Type', 'text/plain')
  res.end('Hello World\n')
})
server.listen(port, hostname, () => {
  console.log(`Server running at http://${hostname}:${port}/`)
})
```

Для того чтобы запустить этот код, сохраните его в файле `server.js` и выполните в

терминале такую команду:

```
node server.js
```

Для проверки сервера откройте какой-нибудь браузер и введите в адресной строке `http://127.0.0.1:3000`, то есть — тот адрес сервера, который будет выведен в консоли после его успешного запуска. Если всё работает как надо — на странице будет выведено «Hello World».