

Node.js



Atul Jain
Naman Gupta

what is javascript?

- nothing to do with java
- a complete language
- moving parts of a web page
- not just `window.open()`

nodejs

**And then came Node.js.
JavaScript on the server,
how cool is that?**

**Node.js is a JavaScript
runtime built on Chrome's
V8 JavaScript engine. Node.
js uses an event-driven, non-
blocking I/O model.**

nodejs

**Node.js is really two things:
a runtime environment and
a library**

why nodejs?

- much easier to setup and faster to code a web app in node.js than any other MVC (models, views, controllers) framework (say, django, flask etc).
- great for single page apps
- Odyssey 15 website was hosted on a five lines of code server. WUT? :O
- node.js is faster, (HELL YEAH!)

node in ONE loc (ONE LINE TO RULE THEM ALL)

server.js

```
console.log("Hello, Byld");
```

terminal

```
$ node server.js
```

```
Hello, Byld
```

```
$
```

event driven

- event loop: listens for events and calls a callback function once an event has been detected
- single thread
- event-driven programming is application flow control that is determined by events or changes in state.

non blocking code (for real)

- `var result = database.query("SELECT * FROM hugetable");
console.log("Hello World");`
- event-driven, asynchronous callbacks, by utilizing an event loop.

non blocking code (for real)

- `var result = database.query("SELECT * FROM hugetable");
console.log("Hello World");`
- event-driven, asynchronous callbacks, by utilizing an event loop.
- `database.query("SELECT * FROM hugetable", function (rows) {
 var result = rows;`
- `});
console.log("Hello World");`
-

npm

Node.js' package ecosystem, npm, is the largest ecosystem of open source libraries in the world.

dependency tree

**Ok, this stuff is boring, right?
Let's write some real stuff.**

We will now create a web application serving a
basic purpose of “file-uploading”

What will our application do?

1. The user should be able to use our web-app on their browser.
2. The user should see a welcome page on <http://localhost:3000/go>
3. On the above web-page, there should be a file-upload form which accepts image files
4. On submitting the form, the user will be redirected to <http://localhost:3000/upload> where user will see the uploaded image

what do we need for this?

1. To server HTTP requests - A HTTP server.
2. Our server will answer differently to requests, depending on the URL we need some kind of router which maps requests to request /handlers.
3. We need request handlers once the server has successfully routed a URL request.
 - a. This request handler should be able to handle the POST data coming through static HTML page from our website
 - b. We have to store the image temporarily on the server storage and show it in the /upload.
4. Once the requested URL is routed to proper route and the incoming data is retrieved from that request, we will show that image on next URL ('/upload')
 - a. This has two

let the coding begin

file upload

middlewares

- `app.use()`
- listens to the requests and runs some code selectively
- example `express.static`

node.js is funny

<https://medium.com/@c2c/nodejs-a-quick-optimization-advice-7353b820c92e>

thanks