10
Server side JavaScript

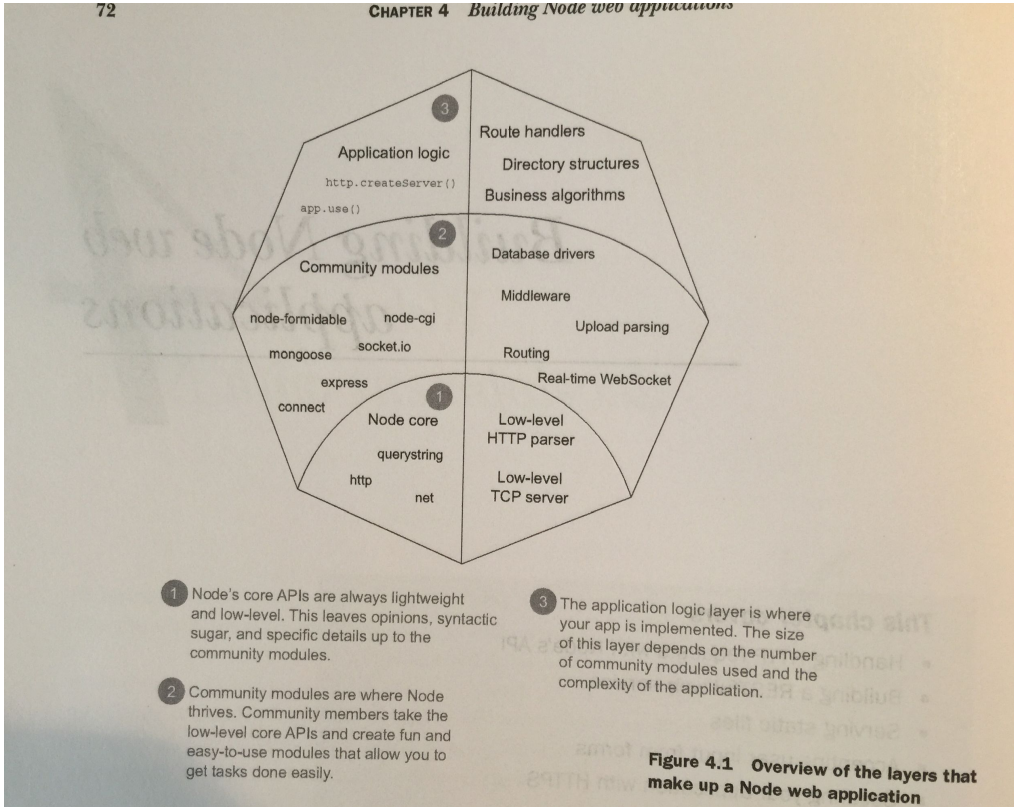
JavaScript is all grown up now :)

- Traditionally, the language was for better or worse bound to the web browser.
- That has all changed in the past 10 years or so.
- JavaScript is everywhere - servers, desktop apps, mobile phone/tablet apps, robots, home automation, IoT devices, etc.
- We will explore a small part of this brave new world of liberated JS by building some applications in JS that run on a web server instead of inside the web browser.

NodeJS

- Node is not really a web application framework, but an asynchronous event driven JavaScript runtime designed for building scalable network applications
- Uses an event loop instead of threading
- Non-blocking IO - unlike PHP/Rails/Django etc NodeJS functions do not block, leading to higher performance due to lower latency
- Created by Ryan Dahl in 2009 - Currently on version 6
- Does not run on top of another web server like PHP runs on Apache
- Capable of real-time streaming applications - video conferencing, multiplayer gaming, remotely controlled hardware like a home security system IoT devices, etc

The software components that make up NodeJS



Concurrency Models

- Concurrent does not necessarily mean multi-threaded.
- Threading - Traditional way of writing concurrent programs. It incurs the cost of context switching and is susceptible to race conditions and deadlocking.
- Event Loop - Seen in many UI applications. Win32 C/C++ programs consisted of around 150 lines of boilerplate code just to setup the “message pump”.
- An event loop will place events (and their callback functions) onto a queue.

JavaScript's Event Loop

- Restaurant analogy - waiter/waitress does not simply wait for cook to complete order for each customer, they take another customer's order while the cook is preparing the food. Thus, the waiter works asynchronously.

- Important to realize that there is an event loop running in the browser's JavaScript environment, and another event loop running in the NodeJS server.

- Each side is processing and responding to events using its own event loop. If you are using Google Chrome then the code doing this is largely the same on both ends (browser and NodeJS both use the V8 JS engine).

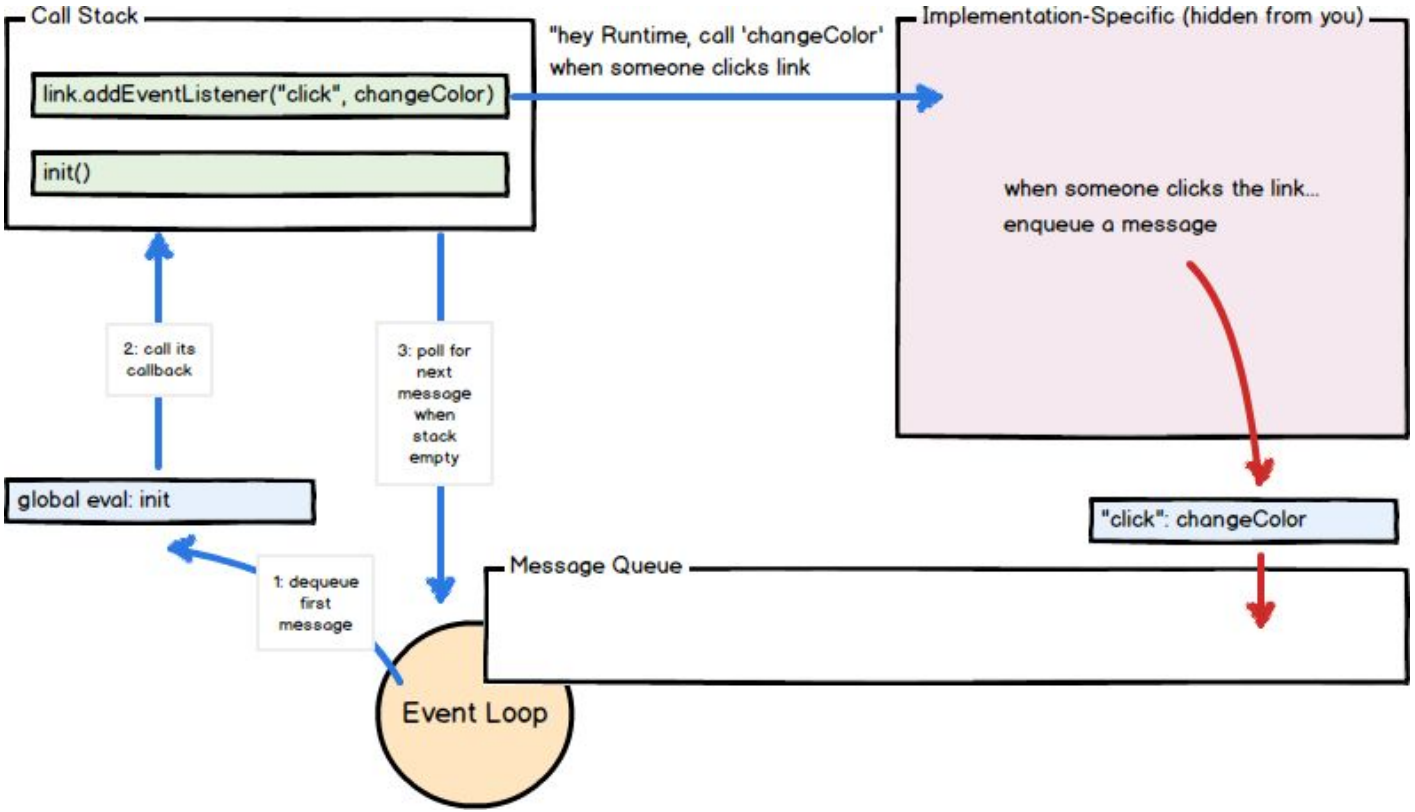
The Event Loop

- Not magic! - just more data structures and algorithms.

- As computer scientists, it is imperative that we understand the mechanics of how these data structures and algorithms work (at least at a conceptual level, if not at a code level) than, say for example, somebody who learned to sling scripts in a 12 week boot camp.

- Under the hood, NodeJS uses the libuv written in C to handle the queuing and processing of asynchronous events.

How the event loop works



Additional Resources

- Official NodeJS site - https://nodejs.org/en/
- Wikipedia page - https://en.wikipedia.org/wiki/Node.js

- High level overview of the JS event loop mechanics - https://www.youtube.com/watch?v=8aGhZQkoFbQ

- Crockford talks about the motivations that led to the creation of NodeJS and why it is designed the way it is with a non-blocking single-threaded event loop 28:00 to about 42:00 is the NodeJS parts - https://www.youtube.com/watch?v=QgwSUtYSUqA

Additional Resources

- Intro to libuv - the library that implements Node's event loop - https://nikhilm.github.io/uvbook/basics.html

- Talk on the basics of libuv - https://www.youtube.com/watch?v=nGn60vDSxQ4

- NPM source - https://github.com/npm/npm
- NodeJS source - https://github.com/nodejs/node
- Libuv source - https://github.com/libuv/libuv
- V8 source mirror - https://github.com/v8/v8

Time for some in-class lab work

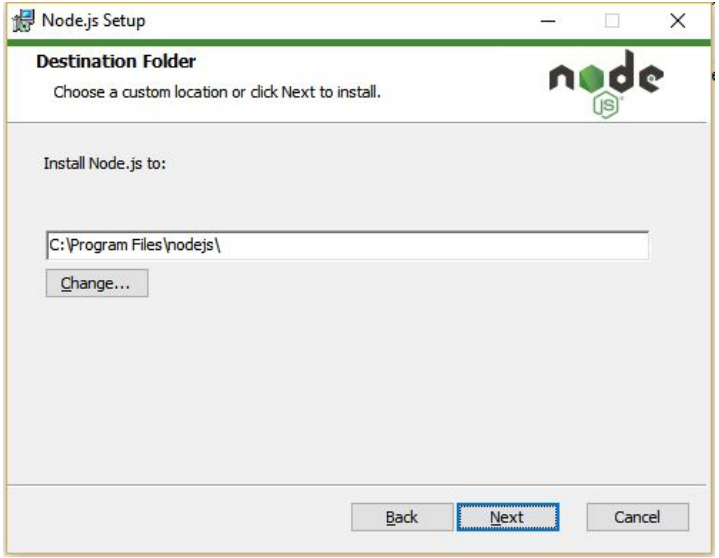
- Download and install NodeJS
- Play around with the Node CLI
- Write a very simple web server in just a few lines of code
- Clone a simple application from GitHub and deploy it to Heroku

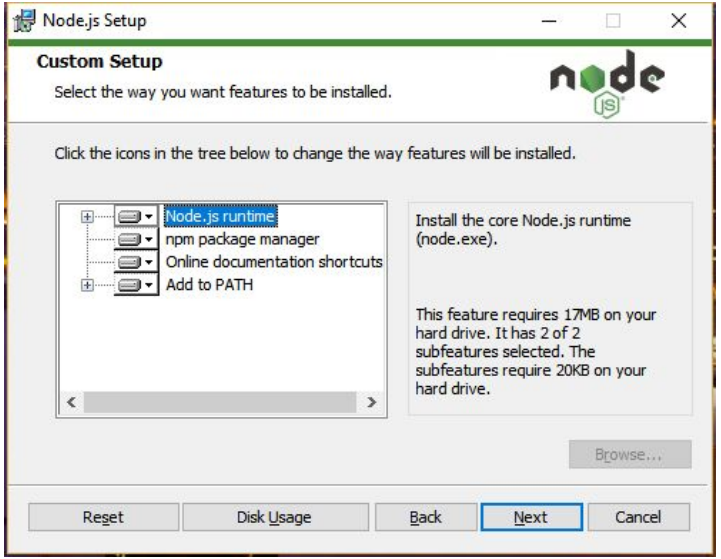
Installing NodeJS on a Mac

- Install XCode command line tools (you don't need to install the whole XCode IDE from the app store) by running "xcode-select --install" in a terminal
- Install homebrew - instructions at https://brew.sh/
- Type 'brew install node'. Homebrew will install node and npm
- Watch this video for more step-by-step instructions if needed (only need to watch first 8 mins) - https://www.youtube.com/watch?v=BIVfpvPnU0U

Installing NodeJS on Windows

- Download installer from https://nodejs.org/en/
- Double-click the installer msi file, mine was called node-v6.10.1-x64.msi
- Follow the prompts accepting all the defaults





Node CLI

- NodeJS has a REPL! - To start it just type ‘node’ at the command prompt
- The global object is where variables defined in the global namespace are attached.
- The console.log also works like it would in the browser, logs messages to the console.

- Node runs under a parent process - usually terminal on a Mac or cmd or PowerShell on Windows. The node command ‘process’ will show this.
- You can make a node module in a .js file and then run it at the CLI using the node command followed by the name of your module.

Let’s run our first real Node app!

- Git clone the repo at https://github.com/heroku/node-js-sample
- Follow the instructions to run the node app locally
- Test the local app is running by navigating to the localhost and port number in your web browser

- Make sure you have the Heroku toolbelt/CLI installed
- Follow the instructions to deploy your app to Heroku
- Test the app by navigating to it in your web browser